IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 29 June 2023, accepted 6 July 2023, date of publication 18 July 2023, date of current version 4 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3296444

RESEARCH ARTICLE

Comparative Analysis of Intrusion Detection Systems and Machine Learning-Based Model Analysis Through Decision Tree

ZAHEDI AZAM, MD. MOTAHARUL ISLAM, AND MOHAMMAD NURUL HUDA
Department of Computer Science and Engineering, United International University, Dhaka 1212, Bangladesh

Corresponding author: Md. Motaharul Islam (motaharul@cse.uiu.ac.bd)

This work was funded by Institute of Advanced Research (IAR) of United International University (UIU) Research Grant Scheme under Grant Ref. No.: IAR-2023-Pub-018.

ABSTRACT Cyber-attacks pose increasing challenges in precisely detecting intrusions, risking data confidentiality, integrity, and availability. This review paper presents recent IDS taxonomy, a comprehensive review of intrusion detection techniques, and commonly used datasets for evaluation. It discusses evasion techniques employed by attackers and the challenges in combating them to enhance network security. Researchers strive to improve IDS by accurately detecting intruders, reducing false positives, and identifying new threats. Machine learning (ML) and deep learning (DL) techniques are adopted in IDS systems, showing potential in efficiently detecting intruders across networks. The paper explores the latest trends and advancements in ML and DL-based network intrusion detection systems (NIDS), including methodology, evaluation metrics, and dataset selection. It emphasizes research obstacles and proposes a future research model to address weaknesses in the methodologies. The decision tree, known for its speed and user-friendliness, is proposed as a model for detecting result anomalies, combining findings from a comparative survey. This research aims to provide insights into building an effective decision tree-based detection framework.

INDEX TERMS Intrusion detection system, machine learning, inductive learning, DDoS attacks, decision tree, supervised and unsupervised learning.

I. INTRODUCTION

Designing intrusion detection systems (IDSs) is greatly challenged by the development of malicious software, commonly called malware. The biggest problem in detecting unknown and disguised malware is that the attackers use various methods to avoid the detection of their activities by the IDS. Consequently, the complexity level of malicious attacks has increased.

Zero-day attacks have greatly affected countries such as Australia and the US [96]. 21st century is seeing more zero-day attacks each year [249], which was higher in volume and intensity than previous years, according to the 2017 Symantec Internet Security Threat Report [225]. The number of data records lost or stolen by hackers has risen to over fourteen billion since 2013 [239], according to the Data Breach Statistics of 2023. Previously, fraudsters targeted bank customers to steal credit cards or bank accounts. But now, the latest malware attacks banks directly, attempting to steal sensitive information in one attack. Thus, the detection of zero-day attacks has gained the utmost importance. The Australian Cyber Security Centre analyzed the complexity of attackers' methods in 2017 [19]. As a result, developing effective intrusion detection systems (IDSs) has become essential to detect new and advanced forms of malware. An IDS aims to identify various malware types replacing the traditional firewall quickly.

Researchers have implemented several ML- and DL-based techniques over the past decade to improve NIDS's ability to identify malicious activities. The tremendous growth in

The associate editor coordinating the review of this manuscript and approving it for publication was Giovanni Pau.

network traffic and ensuing security risks in parallel create difficulties for NIDS systems to identify hostile intrusions effectively. The key idea is to provide up-to-date information on recent ML- and DL-based NIDS to provide a baseline for new researchers exploring this important domain.

Several methodologies employed in signature-based and anomaly-based procedures, such as SIDS and AIDS, are explained. The paper focused on the challenges associated with various anomaly-based intrusion detection methods and their evaluation processes. It then provides recommendations for the most suitable methods based on the type of intrusion.

The discussion highlights how these issues are relevant to the network intrusion detection system research community and how they compare to prior surveys in the field [140], [185].

There is a requirement for a more recent analysis, as earlier surveys on intrusion detection have not thoroughly reviewed dataset issues, evasion techniques, and various attack forms. This work provides a revised classification of the field of intrusion detection, which enhances previous classifications [26], [140].

This and previously published surveys present a detailed overview of the IDS methods and datasets shown in Table 1. According to the detection methodologies, intrusion detection systems were categorized in Axelsson's survey on intrusion detection systems and taxonomy [39]. Liao et al. taxonomy of intrusion systems [140] have classified five subclasses, Statistics-based, Pattern-based, Rule-based, State-based, and Heuristic-based, with an in-depth look at their properties. On the other hand, the signature detection concept, anomaly detection, taxonomy, and datasets are the main topics of our work.

Existing review articles by Buczak and Guven [55], Axelsson [39], Ahmed et al. [26], Lunt [147], and Agrawal and Agrawal [21], concentrate on intrusion detection methods, dataset issues, specific types of computer attacks, and IDS evasion. There is a need for an update because various other intrusion-detection system designs have been developed in the meantime due to the evolution of these systems. The new taxonomy of the intrusion-detection discipline is described in this study and further improves taxonomies provided [26], [140].

The remainder of the essay is structured as follows: it discusses the research strategy used for this investigation. explains the fundamental IDS principle and categorization techniques. The DL and ML methodologies used are described in detail in Section IV. The specifics of the benchmark public datasets and the evaluation measures are illustrated. We give observations, current patterns in NIDS design and various decision tree techniques, research problems, and the future research focus.

This article contributes to the following, given the prior surveys' discussion:

- Categorizing different types of intrusion detection systems based on intrusion methodologies, deployment strategies, and validation strategies allows for a comprehensive comparison of their results and performance, enabling the determination of the most effective model.
- To select the optimal decision tree (DT) based IDS model, a categorization process is conducted, considering intrusion methodologies, deployment strategies, and validation techniques. By evaluating their results and performance, the most appropriate DT model can be identified and implemented.
- We have Highlighted different contemporary efforts to enhance the security of IDS is being addressed by the proposed model, and adaptive Changes have been taken with the existing model to enhance the efficiency of the Detection Rate (DR).
- We have analyzed the performance of datasets. Many existing IDS data sets have been discussed in our paper to understand which data sets are the most state-of-the-art and can be taken as a reference.
- We have addressed the difficulties faced by IDS in the proposed methods. How attackers use different evasion techniques to bypass the detection and inject malicious code.
- Finally, based on our study, we have proposed a model that appears to be the most promising approach for analyzing diverse outcomes.

The following structure describes how the paper is designed. Following the abstract and introduction work procedure is discussed in section II. In section III background study of IDS is described, and computer attack classifications and some recent cyberattacks have been discussed. Section IV describes a review of different classification methods of studies in deep learning and machine learning. Sections V, VI, VII, and VIII illustrate performance metrics, datasets, feature selection, attack types, and evasion techniques related to IDS. Section IX represents research challenges and different Decision Tree (DT) technique analyses. Section X outlines the open research issues that need more focus and improvement. Section XI describes the future works. The article concludes in section XII.

II. WORK PROCEDURE

The research thoroughly investigates ML and DL-based NIDS and decision tree techniques by analyzing existing journal articles. To collect and evaluate relevant information on the topic, a systematic literature review approach is adopted [118]. Two stages of this systematic review were completed. This review article's main goal is to answer the successive queries: (i) What are the latest advancements in the design of AI-based NIDS? (ii) What are the ML and DL methods that have been recently employed for the development of NIDS? What are the advantages and disadvantages of each method that has been adopted? (iv) What are the AI-based NIDS datasets being currently examined? (v) What are the commonly used evaluation metrics for AI-based NIDS? (vi) What can be anticipated in terms of research directions

TABLE 1. Comparison among review article.

Review Article	NIDS Focused	AI Approach		SIDS	Hybrid IDS
		ML	DL		
Vasilomanolakis et al. [235]	X	✓	X	X	X
Lunt et al. [147]	X	X	X	✓	X
Thomas et al. [232]	✓	✓	X	X	X
Liao, et al. [140]	X	X	X	✓	✓
Khraisat et al. [127]	X	✓	X	X	X
Ahmed et al. [25]	X	X	X	✓	X
Buczak et al. [55]	X	✓	X	X	X
Axelsson et al. [39]	X	X	X	✓	X
Liu et al. [144]	✓	✓	✓	X	X
Agrawal and Agrawal et al. [21]	X	X	X	✓	✓
Buczak and Guven et al. [55]	✓	X	X	✓	✓
Da Costa et al. [69]	X	✓	✓	X	X
This articlel	✓	✓	✓	✓	✓

for AI-based NIDS in the future? (vii) DT Techniques in view of ML & DL.

III. BACKGROUND STUDY

A. IDS

The words ''intrusion detection system'' or ''IDS'' are a mashup of the two terms. An intrusion has occurred to disrupt the protection of information stored in computer or network systems, affecting its reliability, privacy, or accessibility [82], [167]. An IDS is a security tool designed to identify unauthorized activity. The system constantly observes the actions of both hosts and networks to identify activity that violates predetermined security protocols, thereby jeopardizing the confidentiality, integrity, and accessibility of information [71], [234]. An IDS will notify the host or network administrators of any malicious activity. Figure 1 shows a passive implementation of a NIDS connectivity.

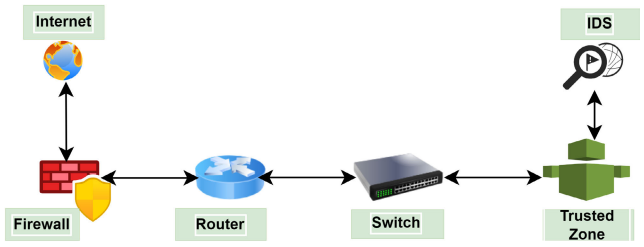

FIGURE 1. Passive implementation of NIDS.

The IDS can be positioned between the network switch and firewall, employing port mirroring technology to monitor incoming and outgoing network traffic. This enables the detection of intrusions.

B. COMPUTER ATTACKS CLASSIFICATIONS

Different classes of cyberattacks exist based on the objectives and targets of the attacker. According to Sung and Mukkamala [223], attack types can be divided into four categories.

The objective of Denial-of-Service (DoS) attacks is to restrict the network or computer services provided to users.

Probing attacks are aimed at gathering information about the network or computer system.

User-to-Root (U2R) attacks aim to acquire root or administrator access to a particular computer or system where the attacker had initial user-level access as a non-privileged user.

Remote-to-Local (R2L) attacks involve sending packets to the targeted machine. Different types of computer attacks fall under these general categories.

C. RECENT CYBER ATTACKS

Over the past two years, social media and smishing attacks have emerged as the primary methods for carrying out social engineering (SE) attacks. These attacks heavily rely on direct engagement between the attacker and the target. In certain instances, SE attacks may involve a basic phone call where the perpetrator pretends to be an employee in order to extract sensitive information like passwords or PIN codes. Phone scams caused Americans a financial loss of around USD 29.8 billion in 2020 [216]. Table 2 presents a comprehensive summary of SE-based attacks and other techniques employed in cyberattacks. The breaches mentioned in table 2 are among the significant security breaches that have occurred in recent years. A combination of human errors and social engineering (SE) attacks contributed to the occurrence of these breaches. The table 2 highlights the importance of human error in the execution of social engineering (SE) attacks. Hackers have the ability to manipulate victims into making mistakes as part of SE attacks or other types of attacks.

TABLE 2. Recent cyber attack analysis.

Attack Event	Year	Description	Technique	Influence Methods	Human Vulnerability
U.S. federal agency [7]	2023	Hackers exploited a vulnerability in the organization's Microsoft IIS server.	Malware	Cyberespionage campaign	Social influence victim
U.S. outpost in Guam [7]	2023	Communications networks were infiltrated by hackers from China.	Spoofing	Complicating the thinking process, curiosity	Trsutful Nature
Finnish Parliament Attack [6]	2022	During a parliamentary session, the website of the Finnish parliament encountered a DDoS attack.	DDOS	Retaliation	Human Influence
Saudi Aramco [8]	2021	The hackers asserted their possession of nearly 1 terabyte of Aramco data and requested a ransom of USD 50M.	Ransomware	Moral influence	Being helpful
European governments [7]	2023	As European governments exhibit a growing readiness to confront China regarding cyber offenses, a spearphishing attack has taken place.	Challenge Opponent	Negligence	Retaliation
Microsoft [9]	2021	Multiple MS Office users were the victim of phishing emails. The victim was scammed for 100-199$l.	BEC attack, Phishing email	Interpersonal deception theory (IDT)	Negligence
Ukraine and other European [7]	2023	The individuals or entities that have offered humanitarian assistance to Ukraine during the war have become targets of attacks.	Ransomware and Supply chain	Reciprocity Norm	Excitement fear
Twitter [5]	2020	Utilized 45 influential accounts to promote a Bitcoin scam.	SE attack, Ransomware	Spear-phishing attacks	Kindness
Technion University, Israel [10]	2023	The university's files were encrypted by hackers who demanded a ransom of 80 bitcoin, equivalent to approximately 1.7M USD, for decryption.	Ransomware	Informative influence	Trying to be acceptable in social norms
Toyota [4]	2019	After falling victim to a BEC attack, Toyota Boshoku Corporation lost USD 37M.	Phishing email (i.e., BEC)	Persuasion using authority	Panic negligence
Attack on Brazilian Banks [1]	2017	Cybercriminals executed an operation by redirecting online traffic of a prominent Brazilian bank to flawlessly replicated fraudulent websites.	DNS Spoofing	Greedy, excitement fear	Greedy
Google and Facebook [2]	2015-23	Phishing emails resulted in financial losses exceeding USD 100 million for both Google and Facebook.	Persuasion using authority/credibility	Framing effect/cognitive bias,	Being obedient to authority
Debt-IN Consultants Cyberattack [6]	2021	Illegitimate access to servers resulted in unauthorized retrieval of client and employee data, including PII.	CIA violation	Ransomware	negligence
Desjardins [3]	2019	2.9M customers data of Desjardins, a Canadian credit union, was compromised & exposed.	Insider Threat	Complicating the thinking process	Curiosity

Upcoming research studies in machine learning and intrusion detection will be explored, introducing novel approaches and diverse applications. Effectively handling evolving network traffic patterns and regularly updating models become crucial tasks to enhance the efficiency of security systems over time. Similar to the importance of providing labelled training data for models, it is equally vital to present this data in a format that models can effectively utilize.

Unmanned Aerial Vehicles (UAVs) are rapidly advancing technologies utilized in military and industrial sectors for various critical tasks such as surveillance and mission control. Whelan et al. [248] introduced a proposed intrusion detection method to address the security risks associated with wireless communication protocols in high-threat environments. The approach aims to mitigate potential threats posed by GPS spoofing and jamming.

The Internet of Things (IoT) technology relies on the concept of interconnected objects exchanging data through Internet connections in our everyday lives. With the growing popularity of this technology, the number of susceptible devices to attacks also rises. Security incidents are a frequent occurrence. However, traditional intrusion detection mechanisms may not effectively function in IoT environments due to the constrained capabilities of IoT devices and the specific protocols they employ.

In their proposed approach, Roy et al. [200] conducted a series of optimizations specifically tailored for IoT networks with limited resources. They achieved favourable outcomes with reduced training data. While big data architectures offer valuable insights through high-level knowledge discovery, managing such data presents its own set of challenges. Traditional information processing technologies struggle to

efficiently handle big data in these architectures and detect network traffic intrusions.

Minimizing false positives in Intrusion Detection Systems (IDSs) is a significant objective. However, when dealing with big data, achieving this goal can be time-consuming due to lengthy training periods. In their work, Ponmalar and Dhanakoti [187] have presented a novel technique aimed at enhancing the intrusion detection process. Their approach specifically tackles the inherent complexities of big data, including various types of non-security data.

IV. HIGH-LEVEL CLASSIFICATION OF INTRUSION DETECTION

IDS can be categorized in terms of how it deploys or detects. The classification taxonomy is given in Figure 2.

A. DEPLOYMENT BASED IDS

From a deployment perspective, IDS can be divided into two categories: Host-based IDS and Network-based IDS [167], [230]. HIDS is installed on single host information, which monitors all activity on that host. Detect any violations of security policies or suspicious behaviour. The main drawback of HIDS is that every host has to be installed that requires intrusion protection, leading to added processing burden on each node, resulting in inefficiency in overall IDS performance [115]. On the contrary, NIDS is deployed within the network to protect against intrusions affecting the network itself and all devices connected. It continuously monitors network traffic, searching for any security violations.

1) NIDS BASED DETECTION TECHNIQUES

Figure 3 shows the three main processes that typically create a NIDS using ML and DL methods: Preprocessing, Training and Testing. This stage usually involves transforming the data into a standardized format and cleaning it if necessary. The process of encoding and normalizing the data is typically performed at this stage. Duplicate entries and entries with missing values are removed.

Preprocessing: Preprocessing in machine learning refers to the steps and techniques applied to raw data before training a machine learning model. It involves transforming and preparing the data to make it suitable for the learning algorithm and to improve the model's performance and accuracy.

The preprocessing phase typically includes the following steps:

1) Data Cleaning: This step involves handling missing values, dealing with outliers, and addressing any inconsistencies or errors in the data. Missing values can be imputed using techniques like mean, median, or interpolation, while outliers can be handled by removing them or transforming them to a more reasonable range.
2) Feature Selection/Extraction: In this step, relevant features are selected or extracted from the available dataset. This can involve removing irrelevant or redundant features to reduce dimensionality and improve computational efficiency. Feature extraction techniques like Principal Component Analysis (PCA) or Singular Value Decomposition (SVD) can be applied to derive new features that capture the most important information in the data.
3) Feature Scaling/Normalization: Machine learning algorithms often perform better when the features are on a similar scale. Scaling techniques like Standardization (mean centering and variance scaling) or Normalization (scaling to a specified range) ensure that features have similar ranges and distributions. This prevents certain features from dominating the learning process based solely on their magnitudes.
4) Handling Categorical Variables: Categorical variables (e.g., gender, color, or country) are typically encoded into numerical representations for machine learning algorithms to process. This can be done through one-hot encoding, where each category is converted into binary variables, or label encoding, where categories are assigned integer labels.
5) Data Splitting: The dataset is divided into training, validation, and testing subsets. The training set is used to train the model, the validation set helps in tuning hyperparameters and evaluating the model's performance during development, and the testing set is used to assess the final model's performance on unseen data.
6) Handling Imbalanced Data (if applicable): In cases where the classes in the dataset are imbalanced, where one class is significantly underrepresented compared to others, techniques such as oversampling, undersampling, or generating synthetic samples can address the class imbalance problem.

Preprocessing is crucial in machine learning as it helps ensure that the data is in a suitable format for training models and can significantly impact the performance and accuracy of the resulting models. Proper preprocessing techniques can improve the model's ability to learn patterns, reduce computational complexity, and prevent biased or misleading results [30].

The original data undergoes preprocessing, after which it is randomly split into two parts: a training dataset, which makes up roughly 80 % of the total data, and a testing dataset, which consists of the remaining 20 % [33], [105]. The ML or DL algorithm is trained using the training dataset in the subsequent training phase. The duration for the algorithm to learn is influenced by the dataset's size and the complexity of the utilized model. Deep Learning models have intricate structures, often requiring a longer training time [137]. Once trained, the model is evaluated using the testing dataset by measuring its accuracy based on the predictions made [16], [25]. For NIDS models, the purpose is to classify network traffic instances as either benign (normal) or attack.

Additionally, Figure 4 shows The classification of recent machine learning and deep learning techniques used for network intrusion detection based on their methodology.

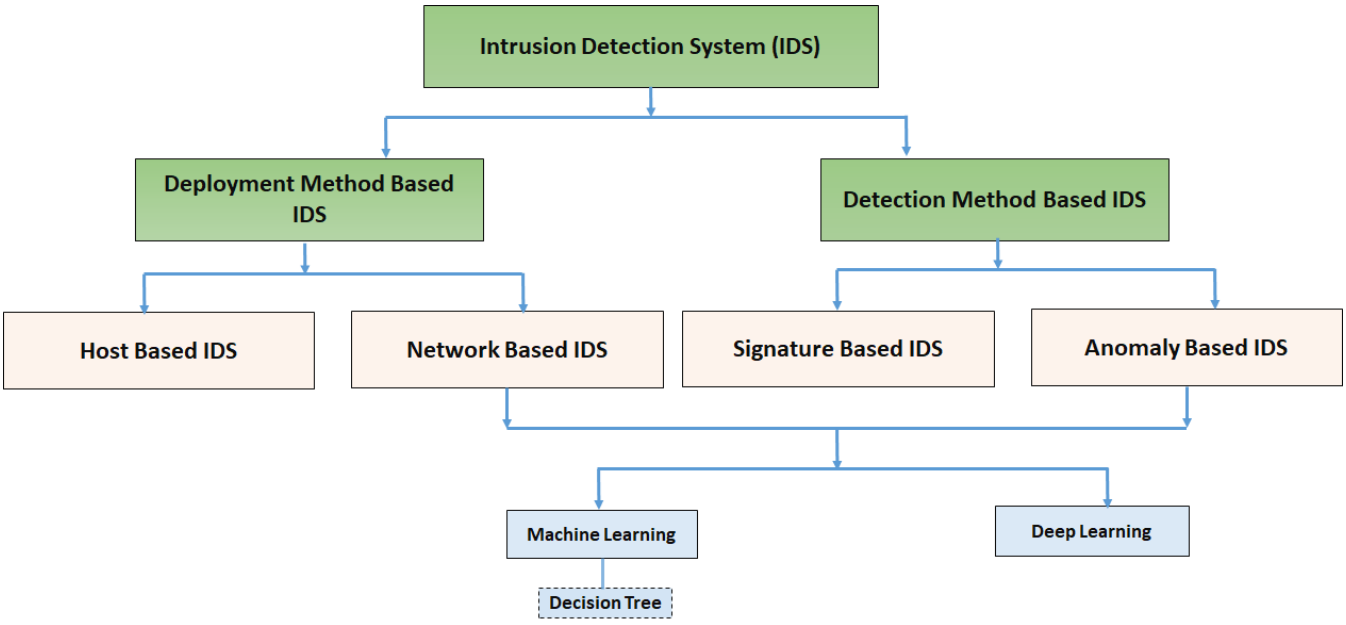

FIGURE 2. Intrusion detection system classification taxonomy.

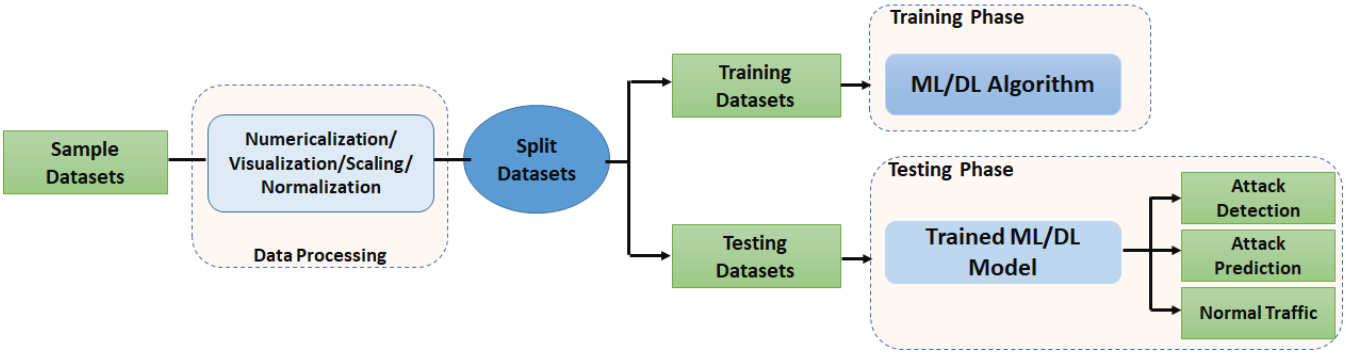

FIGURE 3. Methodology for a network intrusion detection system based on generalized ML/DL techniques.

a: DEEP LEARNING

a.1) Recurrent Neural Networks (RNN): Recurrent Neural Networks (RNNs) are a type of artificial neural network specifically designed for processing sequential data, where the order and context of the data elements are crucial. Unlike feedforward neural networks, which process data in a single pass from input to output, RNNs have a recurrent connection that allows them to retain and utilize information from previous steps or time points in the sequence.

The fundamental building block of an RNN is the recurrent layer, which contains recurrent units or cells. Each cell maintains an internal state, or ''memory,'' that is updated and passed along to the next step in the sequence. This memory serves as a context, allowing the network to capture dependencies and patterns across different time steps.

The input is combined with the recurrent unit's previous hidden state (output) at each time step. This combined information is then passed through an activation function, such as the hyperbolic tangent (tanh) or rectified linear unit (ReLU), to produce the current hidden state. The hidden state at each time step can be considered as an encoding of the current input and the previous context.

One of the key advantages of RNNs is their ability to handle input sequences of variable length. They can process input sequences of any length by unrolling the recurrent layer for the desired number of time steps. This flexibility makes RNNs well-suited for natural language processing, speech recognition, machine translation, sentiment analysis, and time series prediction.

However, standard RNNs suffer from the ''vanishing gradient'' problem, where the gradients used for learning are prone to diminishing exponentially over time, making it difficult to capture long-term dependencies. Various RNN variants have been developed to address this issue, such as Long

Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), which incorporate gating mechanisms to preserve better and update the memory state.

LSTM and GRU cells are designed to selectively retain and forget information, allowing them to capture long-range dependencies more effectively. These variants have become widely used and have demonstrated superior performance in many sequence modelling tasks.

RNNs are neural networks with recurrent connections that enable them to process sequential data by maintaining and utilizing contextual information. They have proven powerful tools for modelling and predicting sequential patterns in various domains. RNNs find application [161] in several fields, including speech processing, human activity recognition, prediction of handwriting, and semantic understanding [86], [88] [158], [177].

In IDS, RNN is usually applied for supervised classification and feature extraction [145]. Some examples of RNN variants such as Long short-term memory (LSTM) [48], gated recurrent unit (GRU) [66], [163] are proposed to solve these issues. Recently, Chung et al. [65] introduced the Gated Feedback Recurrent Neural Network (GF-RNN) to address the issue of learning at a multiplicative level. This is achieved by adapting the previous hidden state and assigning different layers with varying time scales.

Yin et al. proposed using an RNN-based intrusion detection system for the multi-class and binary classification of the NSL-KDD dataset [261]. The model's performance was evaluated by testing it with varying numbers of hidden nodes and learning rates, and it was concluded that both parameters impacted the model's accuracy. Optimal outcomes were observed when utilizing 80 hidden nodes and either a learning rate of 0.1 or 0.5, depending on whether the situation was binary or multi-class. The suggested model performed well compared with ML methods, and a smaller RNN model was reported in [211]. The key flaw in this research is the increased computational processing, which increases the training time of the model and decreases detection rates for U2L and R2R classes. The performance evaluation of the suggested model against several other DL approaches is also missing from the article.

Xu et al. presented an RNN-based intrusion detection system that employed a multilayer perceptron, a softmax classifier, and a GRU cell as the central memory component in [254] using NSL-KDD and KDD Cup'99 datasets. Xu et al.'s approach showed promising results compared to other methodologies, but it has a limitation in detecting minority attack classes such as U2R and R2L, leading to lower detection rates for these classes.

Naseer et al. suggested a comparison study of intrusion detection systems using several deep learning and machine learning algorithms, with the implementation taking place on a testbed that utilized a GPU [173]. Naseer et al. used a dataset for benchmarking purposes. The experimental results indicated that the LSTM and Deep CNN models performed better in accuracy than other models.

a.2) AutoEncoder: AutoEncoder (AE) is a well-known deep learning technique that falls under the category of unsupervised neural networks [76]. An autoencoder is an artificial neural network that tries to recreate the input as closely as possible by learning the most relevant features. The dimensions of the hidden layers are usually smaller than the input layer, but it has an input layer and an output layer of the same size, and it operates in an encoder-decoder fashion. Different types of autoencoders include Stacked AE, Sparse AE, and Variational AE [84].

Papmartizivanous et al. [183] suggested an automated misuse detection system that employs the benefits of self-taught learning through proper alignment [194], MAPE-K frameworks [120]. The authors utilized a sparse autoencoder (AE) as the unsupervised learning algorithm in the Plan activity of the MAPE-K framework. They applied it to the NSL-KDD and KDD Cup'99 datasets to extract useful features. The limitation of this approach was its reduced precision when detecting U2R and R2L attack categories.

Shone et al. [215] presented an intrusion detection system that combined deep autoencoders (AE) with a machine learning technique called random forest (RF). The model was created with the objective of improving computational and temporal efficiency by solely utilizing the encoder component of the AE. The system consisted of two stacked, nonsymmetric deep autoencoders with three hidden layers each, and RF was used for classification. Experiments were conducted on the KDD Cup '99 and NSL-KDD datasets for multiclass classification scenarios. Compared to the Deep Belief Network (DBN) mentioned in [32], although this model shows enhanced detection accuracy and quicker training duration, it is limited by its inadequate training data for R2L and U2R attacks, leading to weak detection of these types of attacks.

In their research, A-Qatf et al. [28] introduced a self-taught learning technique that involves using sparse autoencoders and support vector machines (SVMs). By conducting experiments on the NSL-KDD dataset, they demonstrated enhanced overall performance of the proposed model. However, the performance of the proposed method for the R2L and U2R classes was not reported.

Yan and Han [256] proposed an intrusion detection system that combined a stacked sparse autoencoder (SSAE) with support vector machines (SVM). The SSAE extracted features, and the SVM was used as the classifier. The model was evaluated for both binary-class and multi-class classification using the NSL-KDD dataset, and the results showed that the proposed model outperformed other feature selection, machine learning, and deep learning techniques. Although the model performed reasonably well in detecting U2R and R2L attacks, its detection rates for these classes were still lower than those in the dataset.

A two-stage model that utilizes deep stacked autoencoders (AEs) was suggested as an effective approach by Khan et al. [121]. The dataset was first divided into the attack and normal classes using probabilities. The final step of classifying normal and multiclass attacks used the probability scores as an additional feature in decision-making. The performance of the model was assessed on two datasets, KDD Cup'99 and UNSWNB15. The datasets were downsampled to address the issue of duplicate records, and SMOTE was used to balance the record distribution in the UNSWNB15 dataset. These preprocessing techniques were applied to improve the detection rate efficiency of attack classes with lower training instances.

Malaiya et al. proposed different IDS models using fully connected networks, variational autoencoders, and sequence-to-sequence (Seq2Seq) architectures [151]. These models were tested using several datasets, including NSL-KDD, Kyoto Honeypot, UNSW-NB15, IDS2017, and MAWILab traces [79]. Among the proposed models, the Seq2Seq model, constructed using two RNNs, achieved the best detection accuracy compared to the other models across all the datasets.

Yang et al. suggested an intrusion detection system (IDS) model that employs a deep neural network, supervised adversarial variational autoencoder, and regularization [257]. The performance of SAVAER-DNN was evaluated using two benchmark datasets, NSL-KDD and UNSW-NB15.

Andresini et al. [37] proposed a multistage ID model that incorporated the concept of AutoEncoder (AE). The proposed model included a convolutional layer and two fully connected layers. During the first unsupervised phase, two separate autoencoders were trained on Normal and Attack data to reconstruct the samples. In the supervised stage, these newly reconstructed samples were used to create a new augmented dataset as input to a 1D-CNN. The output of the convolution layer was flattened and fed to the fully connected layers, which then used a softmax layer to classify the dataset. The effectiveness of the proposed approach was assessed using KDD Cup'99, UNSWNB15, and CICID2017 datasets and demonstrated superior performance compared to other deep learning models. However, it does not offer any insight into the attack characteristics.

a.3) Deep Neural Network (DNN): The fundamental structure of Deep Neural Networks (DNNs), which includes an input layer, an output layer, and multiple hidden layers, the model can learn in multiple layers. For intricate nonlinear functions, DNN is used. The model's capability is improved by an increase in the number of hidden layers, which raises the model's abstraction level [87]. Jia et al. [111] proposed classifying the datasets KDD cup'99 and NSL-KDD, a network IDS based on DNN with four hidden layers. The output layer included one fully connected layer and a softmax classifier for classification. The activation function for the hidden layer was a rectified linear unit [70] performance evaluation of the proposed model indicated that it is robust and improved the detection rates for most of the attack classes except for U2R due to the limited number of records available. The use of optimization algorithms and automatic tuning can help reduce the complexity of the model structure, which in turn reduces the computation time.

Su and Jung [221] described the DNN-based IDS with adversaries and evaluated it using the NSL-KDD dataset. Attacks such as FGSM [85], JSMA [184], DeepFool [165], and CW produced the adversarial samples [60]. The findings indicate that DL-based IDS is more sensitive to the commonly used attributes and requires more attention to safeguard the network against potential attacks.

Vinayakumar et al. [237] introduced a hybrid scalable DNN framework called scale-hybrid-IDS-AlertNet for intrusion detection at the host and network levels. The scalable platform was implemented using Apache Spark cluster computing platform [238]. The effectiveness of the suggested NIDS model was tested on various publicly available datasets, such as KDDCup 99, NSL-KDD, Kyoto, UNSW-NB15, WSN-DS, and CICIDS 2017. The experimental results showed that the proposed model outperformed other machine learning algorithms.

a.4) Deep Belief Network (DBN): A Deep Belief Network (DBN) is a DL model that consists of multiple Restricted Boltzmann Machines (RBM) stacked together and a softmax layer for classification purposes [97]. In DBN, multiple Restricted Boltzmann Machines (RBMs) are stacked to form a deep network, and a softmax layer is added for classification. RBMs are bi-directional models with connections between input and hidden layers. DBN utilizes a layer-by-layer training approach, first pre-training the network in an unsupervised manner, then fine-tuning it in a supervised way to extract valuable features [99]. DBN is used in IDS for tasks like feature extraction and classification.

Marir et al. [153] proposed a large-scale network intrusion detection model that utilizes the DBN and a multilayer ensemble SVM on Apache Spark. The proposed approach involved extracting features using DBN, passing them to ensemble SVM, and making predictions through a voting mechanism. The effectiveness of the method was tested using datasets from KDD Cup'99, NSL-KDD, UNSW-NB15, and CICID2017, and it demonstrated outstanding performance in detecting abnormal activities in a distributed fashion.

Wei et al. [247] proposed a DL-based model called DBN that was optimized using a combination of particle swarm, fish swarm, and genetic algorithms to improve the accuracy of intrusion detection systems. The NSL-KDD dataset was used to test the model, and it demonstrated notable enhancements in detecting the U2R and R2L classes. Nonetheless, the model's complicated structure resulted in a longer training time, which is a disadvantage.

a.5) Convolutional Neural Network (CNN): The Convolutional Neural Network (CNN) is an ideal structure for handling data in the form of arrays. It consists of an input layer, followed by a sequence of convolutional and pooling layers for feature extraction, a fully connected layer, and a softmax

classifier in the classification layer. CNNs have proven highly effective in the field of computer vision [134], which are used for supervised classification and feature extraction.

Yihan et al. [260] introduced CNN based on IDS. Initially, the algorithm employs Principle Component Analysis and Autoencoder for feature extraction. The resulting one-dimensional feature set is converted into a two-dimensional matrix and provided as input to the Convolutional Neural Network. The model's performance was evaluated on the KDD Cup'99 dataset, and the experimental results confirmed its effectiveness, particularly regarding its training and testing times. However, there is a drawback, as the detection rates for U2R and R2L attacks are lower than others.

Jiang et al. [112] introduced a deep hierarchy-based Intrusion Detection System (IDS) that incorporates CNN and bidirectional long short-term memory (BiLSTM). The class imbalance problem was addressed using SMOTE to increase the minority samples, which helped the model fully learn the features. The BiLSTM was utilized to extract temporal features, while the CNN was employed to extract spatial features. The experiments were conducted using datasets from UNSWNB15 and NSL-KDD. The results showed that the proposed method improved the accuracy and detection rate.

Zhang et al. [265] proposed a sophisticated Intrusion Detection System (IDS) model that combines CNN and gcForest, using the P-Zigzag algorithm to convert raw data into two-dimensional grayscale images. For preliminary detection, they employed a coarser layer of a more advanced CNN model (GoogLeNetNP). The abnormal classes were further divided into N-1 subclasses using gcForest (caXGBoost) in the fine-grained layer. The researchers used a dataset created by merging the CIC-IDS2017 and UNSW-NB15 datasets. The results obtained from the experiment demonstrate that the proposed model effectively reduces the rate of False Alarms and, at the same time, surpasses individual algorithms in terms of precision and detection rate.

Xu et al. [254] presented an Intrusion Detection System (IDS) model based on the novel deep learning concept of Few-shot Learning (FSL). The approach trains the model using a small portion of the dataset's balanced labelled data. The model utilized DNN and CNN as embedding functions to extract important features and decrease dimensionality and was evaluated on the NSL-KDD and UNSW-NB15 datasets. The findings indicated that the model achieved acceptable detection rates for attack classes with fewer instances.

In this section, we have recognized a range of Machine Learning (ML) and Deep Learning (DL) methods summarized in Table 3, which researchers have recently developed to find network intruders. Table 4 also highlights the benefits and drawbacks of each methodology. These methods must, however, be assessed using specific metrics.

a.6) Fully Connected Neural Networks (FCNN): Fully Connected Neural Networks (FCNN), also known as dense networks or multilayer perceptrons (MLPs), are a fundamental type of artificial neural network used in machine learning and deep learning. FCNNs consist of multiple layers of interconnected nodes called neurons or units, with each neuron in a layer connected to every neuron in the subsequent layer.

The structure of an FCNN typically consists of an input layer, one or more hidden layers, and an output layer. The input layer receives the input data, which could be a vector representing features or pixels of an image. The hidden layers perform intermediate computations, and the output layer produces the final prediction or output.

Each neuron in an FCNN computes a weighted sum of the inputs it receives, applies an activation function to the sum, and passes the result to the next layer. The weights represent the strength or importance of the connections between neurons, which are learned during training. Bias terms are often added to each neuron to introduce more freedom.

Activation functions play a crucial role in FCNNs as they introduce non-linearity into the model, enabling the network to learn complex patterns and relationships in the data. Common activation functions used in FCNNs include the sigmoid function, ReLU (Rectified Linear Unit), and softmax (for multi-class classification).

The learning process of an FCNN involves two main steps: forward propagation and backpropagation. During forward propagation, the input data is fed into the network, and the activations of neurons are computed layer by layer until the output layer produces the predicted output. Backpropagation is then used to compute the gradients of the network's parameters (weights and biases) with respect to a given loss function. These gradients are used to update the parameters through optimization algorithms like gradient descent, aiming to minimize the loss and improve the model's performance.

FCNNs are powerful models capable of learning complex non-linear relationships in data. They have been successfully applied to various tasks, such as image classification, natural language processing, and time series analysis. However, overfitting can be a challenge as the number of parameters in FCNNs grows with the number of neurons and layers. Regularization techniques, such as dropout or L2 regularization, are often employed to mitigate overfitting.

Overall, FCNNs provide a flexible framework for learning from data, allowing the extraction of intricate patterns and making them a key component in deep learning. The feed-forward aspect of the network means that the neurons in any layer do not connect to neurons in a layer below them. This architecture is commonly used for feature extraction [245].

a.7) Generative Adversarial Networks (GAN): Generative Adversarial Networks (GANs) are a class of machine learning models that consist of two neural networks: a generator network and a discriminator network. GANs are used for unsupervised learning and are particularly powerful in generating realistic synthetic data, such as images, text, and audio.

The primary objective of GANs is to train the generator network to produce synthetic data that is indistinguishable from real data, while the discriminator network aims to

TABLE 3. Comparison among various ML and DL techniques.

Article	Algorithm		Technique
	ML	DL	
Yin et al. [263]	X	✓	Recurrent Neural Network
J. Gao et al., Sarkar et al. [80] [206]	X	✓	CNN, RBM
Shen et al. [212]	✓	X	Ensemble Method with Optimization using BAT algorithm
Bhattacharya, Lane [47]	X	✓	Sparse coding and Convolutional Neural Networks
Shone et al. [215]	✓	✓	Non Symmetric Deep Auto Encoder with Random Forest
Khan , Taati [123]	X	✓	Ensemble of Channel-wise Autoencoder
Ali et al. [108]	✓	X	Fast Learning Network and Particle Swarm Algorithm
Ijjina , Mohan [104]	X	✓	Ensemble of Deep Convolutional Neural Networks
Jia et al. [111]	X	✓	Deep Neural Network
Lin et al., Ravi et al. [142] [196] [197]	X	✓	CNN, Conventional feature
Kotpalliwar et al. [227]	✓	✓	RBF-SVM
Wang et al. [221]	X	✓	Deep Neural Network
Chandrasekhar et al. [198]	✓	✓	C-SVM
Alzantot et al. [34]	X	✓	LSTM, Mixture Density Network
Ding and Yuxin et al. [106]	✓	✓	DBN
Yan et al. [258]	✓	✓	Sparse Auto Encoder with SVM
Zhao et al. [164]	✓	✓	DBN-PNN
Naseer et al. [173]	✓	✓	Comparison between different ML and DL-based IDS Models
Yin and Long et al. [172]	✓	✓	RNN
Xu et al. [256]	X	✓	Neural Network using GRU as memory with Multiple Layer Perception
Staudemeyer et al. [271]	✓	✓	LSTM
Al-Qatf et al. [28]	✓	✓	Self Taught Learning Model based on Sparse Auto Encoder and SVM
Kolosnjaji et al. [129]	✓	✓	CNN
Marir et al. [153]	✓	✓	Deep Belief Network and SVM
Papamartizivanous et al. [183]	X	✓	Self Taught Learning based on Sparse Auto Encoder
Khan et al. [121]	X	✓	Two-Stage Model using Stacked Auto Encoder
Xiao et al. [255]	X	✓	CNN with PCA and AutoEncoder for dimension reduction
Yao et al. [260]	✓	X	A Multilevel Model based on K-Means Clustering and Random Forest
Vinayakumar et al. [239]	X	✓	Deep Neural Network
Gao et al. [81]	✓	X	Ensemble Machine Learning methods with Voting algorithm
Wei et al. [249]	X	✓	Deep Belief Network along with optimization algorithms Particle Swarm, Fish Swarm and Genetic Algorithms.
Zhang et al. [267]	X	✓	Multi-Layer Convolutional Neural Network
Malaiya et al. [151]	X	✓	Model based on Fully Connected Networks (FCNs), Variational AutoEncoder (VAE), and Sequence-to-Sequence (Seq2Seq) structures
Karatas et al. [117]	✓	X	Performance Comparison of different ML algorithm by first reducing the dataset imbalance ratio using (SMOTE)
Jiang et al. [112]	X	✓	Deep Hierarchical Network based on CNN and BiLSTM
Yang et al. [259]	X	✓	Supervised adversarial Variational Auto Encoder with regularization (SAVAER) and Deep Neural Network (DNN)
Yu et al. [256]	X	✓	Few Shot Learning (FSL) using DNN and CNN
Andresini et al. [37]	X	✓	Multistage Auto Encoder and CNN

differentiate between real and generated data correctly. The generator network generates samples from random noise, attempting to fool the discriminator network, while the discriminator network learns to distinguish between real and fake data.

Here's a step-by-step overview of how GANs work:

1) Initialization: The generator and discriminator networks are initialized with random weights.
2) Training Process:
   a) Generator Training: The generator network inputs random noise and generates synthetic data samples. The generated samples are then fed into the discriminator network.
   b) Discriminator Training: The discriminator network receives real and generated samples as input and learns to classify them as real or fake. It is trained using a combination of real and generated data with corresponding labels.

3) Adversarial Training: The generator and discriminator networks are trained alternatively in a competitive manner. The generator tries to improve its generated samples to deceive the discriminator, while the discriminator aims to enhance its ability to differentiate between real and generated samples.
4) Convergence: The GAN training process continues until a certain stopping criterion is met or until the generator produces synthetic samples that are close to indistinguishable from real data, as determined by the discriminator.

The adversarial training between the generator and discriminator networks allows the GAN to learn the underlying distribution of the training data and generate new samples that follow a similar distribution. GANs have been successful in various domains, including image generation, text generation, and video synthesis. They have been used for tasks such as image translation, data augmentation, and even generating deepfake videos [20].

However, training GANs can be challenging. It requires balancing the training of the generator and discriminator networks, avoiding issues such as mode collapse (where the generator only produces limited types of samples) and training instability. Various techniques, such as different loss functions, regularization methods, and architectural modifications, have been developed to address these challenges and improve the stability and quality of GAN training.

Bau et al. present an analytical framework [42] to visualize and comprehend GANs at the unit, object, and scene levels. They first used a segmentation-based network dissection technique to pinpoint a collection of comprehensible units closely linked to object concepts. Then, we measure the capacity of interventions to control output objects to quantify the causal effect of interpretable units.

a.8) Restricted Boltzmann Machine (RBM): The restricted Boltzmann machine (RBM) is a generative model [78], [98] used in the training of deep neural networks through greedy layer-by-layer feature learning. It is a type of generative stochastic artificial neural network that belongs to the family of unsupervised learning algorithms. RBMs are widely used in machine and deep learning for tasks such as dimensionality reduction, feature learning, and collaborative filtering.

An RBM consists of two layers: a visible layer and a hidden layer. The visible layer represents the input data, while the hidden layer captures latent or hidden features. The layers are fully connected, meaning each neuron in one layer is connected to every neuron in the other layer, but there are no connections within the same layer.

The main idea behind RBMs is to model the joint probability distribution between the visible and hidden layers using an energy-based approach. RBMs are trained to find the parameters that minimize the network's energy. The weights and biases of the connections between the visible and hidden units define the energy of an RBM.

During training, RBMs utilize a process called contrastive divergence. This process involves two main steps:

1) Positive Phase: The RBM is presented with input data, and the activations of the hidden units are computed based on the inputs. This step captures the correlations between the visible and hidden layers.
2) Negative Phase: The activations of the hidden units obtained in the positive phase are used to reconstruct the visible layer. Then, the activations of the hidden units are again computed based on the reconstructed visible layer. This step captures the reconstructions and helps adjust the weights and biases of the RBM to minimize the energy.

The training process continues iteratively, with the RBM adjusting its parameters to improve the reconstruction of the input data. Once training is complete, RBMs can be used for various tasks. For example, in dimensionality reduction, the hidden layer activations can be treated as a compressed input data representation. RBMs can also be stacked together to form deep belief networks (DBNs) or used in generative models to generate new samples that resemble the training data.

Overall, RBMs provide a powerful tool for unsupervised learning, enabling the discovery of hidden patterns and learning useful data representations.

a.9) Deep Boltzmann Machine (DBM): A generative model called Deep Boltzmann Machine (DBM) [204] has hidden layers with undirected connections throughout the network. It is a type of generative deep learning model that consists of multiple layers of hidden units. It is based on the Boltzmann Machine, which is a type of stochastic neural network. DBMs are designed to capture complex patterns and dependencies in high-dimensional data by learning a hierarchical representation of the input.

DBMs are composed of visible units, which represent the input data, and hidden units, which capture the latent factors or features of the data. The connections between the units are undirected and have weights that determine the strength of the interaction between them. These weights are learned through a training process that aims to maximize the likelihood of generating the training data.

The learning in DBMs is typically performed in an unsupervised manner using a technique called Contrastive Divergence. This learning algorithm iteratively updates the weights based on the difference between the observed data and the generated samples. It involves two main steps: the positive phase and the negative phase. In the positive phase, the DBM is initialized with the input data, and the activations of the hidden units are computed. In the negative phase, the hidden activations are sampled, and the model generates reconstructed samples by iteratively updating the visible and hidden units. The difference between the generated samples and the observed data is then used to update the weights.

DBMs are known for capturing complex dependencies and generating realistic samples from the learned distribution. However, training DBMs can be computationally challenging due to the intractability of computing the partition function,

which is required for efficient learning. As a result, various approximation techniques and learning algorithms, such as Contrastive Divergence, have been developed to overcome these challenges and make training DBMs feasible.

DBMs have been used in various applications, including image generation, feature learning, and collaborative filtering. They are particularly effective when dealing with high-dimensional and complex data, as they can learn hierarchical representations that capture local and global patterns. The model uses Markov random fields for layer-by-layer pre-training on large amounts of unlabeled data and provides feedback using a bottom-up approach similar to Deep Belief Networks (DBN). The algorithm is fine-tuned using back-propagation.

a.10) Sparse Coding: Olshausen and Field [180] introduced sparse coding for the first time as a machine learning technique for learning over-complete basis to create effective data representation reducing the dimensional of data and dynamically represent the data as a linear combination of basis vectors. The data structure is captured by this efficient sparse coding model, which also finds correlations between different input vectors [90]. It is a technique used in machine learning and deep learning for learning efficient and compact representations of data. It aims to find a sparse set of coefficients to reconstruct the input data using a dictionary of basis functions or atoms.

The input data is typically represented as a high-dimensional vector in sparse coding. The goal is to find a sparse representation of this vector by selecting a small number of coefficients while minimizing the reconstruction error. The idea is that by using a sparse set of coefficients, we can capture the essential features of the data while discarding the less critical or redundant information.

The process of sparse coding involves two main steps:

1) Dictionary Learning: The first step is to learn a dictionary or a set of basis functions that form the building blocks for the sparse representation. The dictionary is typically overcomplete, meaning it has more atoms than the dimensionality of the input data. The dictionary can be learned using techniques such as K-SVD, which iteratively updates the dictionary and the sparse coefficients.
2) Sparse Coding: Once the dictionary is learned, the next step is to find the sparse coefficients that best represent the input data. This is typically done by solving an optimization problem, such as an ℓ_1-norm minimization problem, where the objective is to find the sparsest representation to reconstruct the input data with minimal error. Various optimization algorithms, such as the LASSO (Least Absolute Shrinkage and Selection Operator), can solve this problem efficiently.

The sparsity constraint in sparse coding encourages the coefficients to be mostly zero, resulting in a sparse representation. This sparse representation can benefit various tasks, such as denoising, compression, feature selection, and anomaly detection. Sparse coding has been successfully applied in several domains, including computer vision, natural language processing, and signal processing.

In machine learning and deep learning, sparse coding has been used as a component in models such as sparse autoencoders and sparse coding neural networks. These models leverage sparse coding to learn more compact and meaningful data representations, which can improve generalization, reduce overfitting, and provide interpretable features.

Recent studies to learn data representation, particularly in human activity recognition, including the shift in variant method [240] and sparse fusion [72] reducing computational complexities for implementation of human activity recognition system using a mobile phone and wearable devices.

B. DETECTION BASED IDS

The Intrusion Detection System (IDS) can be categorized into two types of detection methods: Signature-based intrusion detection (SIDS) and Anomaly detection-based intrusion detection (AIDS). SIDS is also referred to as ''knowledge-based intrusion detection'' or ''misuse intrusion detection'' and is based on creating a signature for identifying known attack patterns. Attack detection is performed by comparing the data patterns with the stored signatures kept in a signature database [27], [39]. One advantage is that it efficiently detects known attacks since their signatures are easily accessible. A limitation of SIDS is that it cannot detect or identify novel or unfamiliar attacks because no established signature patterns are available. This method is also resource-intensive because a large signature database must be maintained and checked against the data packets for potential intrusions [232].

The idea behind AIDS also referred to as ''behaviour-based IDS,'' is to establish a clear profile of normal behaviour. Any deviation from this typical behaviour is considered an anomaly and will be detected as an attack [148], [174]. The main benefits of AIDS are its capability to identify novel and undiscovered attacks [268] and the fact that the normal activity profile is tailored to specific networks and applications [89]. The main disadvantage of AIDS is the false alarm rate (FAR), making it difficult to distinguish between normal and abnormal intrusion detection patterns [190].

The adoption of IoT devices has increased exponentially due to the popularity of the IoT device and the development of network technology [22], [23]. The Wireless Sensor Network (WSN), composed of multiple sensor nodes for gathering data, is a crucial technology in building an Internet of Things (IoT) network [155]. Here IoT sensor devices gather a significant amount of important data and then distribute it online [209]. There are security challenges for the IoT network because of the complex structure of WSN, which is composed of resource-constrained sensor nodes [95], [135]. IDS is therefore recognized as one of the most effective WSN and IoT security solutions. There are various approaches in the literature for IDS techniques that actually make use of watchdogs, trust models, and game-theoretic concepts.

Network nodes that have been tasked with keeping an eye on and monitoring the network activity of their neighbours are known as watchdogs. Then, using a set of rules, a decision is taken regarding the misbehaving nodes. In the areas of WSN [199], AdHoc [101] networks, and IoT [131], numerous solutions are put forth for anomaly and intrusion detection employing watchdogs.

Another approach for enhancing an IDS's performance is trust models [64]. A trust-based IDS detects malicious nodes by regularly scanning network traffic for unusual behaviour and assessing the nodes' trustworthiness. Watchdog, Bayesian [156], and game theory-based trust models are some of the trust models used in various IDS implementations [13], [213]. To lessen the computational burden on sensor nodes with limited resources in the IoT, it is possible to use a distributed trust management strategy [24], [124].

Game theory is commonly utilized in the design of IDS to improve its effectiveness. Game theory is a mathematical framework that deals with the strategic interactions between multiple players, where each player has a set of strategies and an action plan. Payment is assigned for each decision taken by the players. The adjustment that the player generates to maximize the reward is the foundation for the game's solution. The game can be cooperative or non-cooperative depending on how entities interact cooperatively or competitively. IDS sees the game between attackers and defenders for IoT and WSN as being simulated either via their interaction or the use of an attacker's forecasting approach [12], [13], [14], [15].

Upcoming section XI focuses on reviewing an AI-powered network intrusion detection system (NIDS) that monitors incoming network traffic through an edge router to secure IoT networks. Section XI overviews the most widely used AI-based methods for creating efficient network intrusion detection systems over the past years.

1) AIDS IMPLEMENTATION OVERVIEW

The main types of AIDS techniques are described as machine learning-based [55], [157]; statistics-based [143]; and knowledge-based [75], [59].

a: STATISTICS-BASED TECHNIQUE

The statistical-based method collects and analyzes each data record in a cluster of items to build a regular user behaviour statistical model. Usually, one of the following models is used by statistical IDSs.

a.1) Univariate: In the context of statistical Intrusion Detection Systems (IDS), ''univariate'' refers to the analysis or modelling of a single variable or feature at a time. The univariate analysis focuses on understanding and characterizing individual variables' statistical properties, distributions, and patterns in isolation, without considering their relationships with other variables.

In statistical IDS, univariate analysis is crucial in assessing the behaviour and characteristics of individual network traffic or system metrics. It involves analyzing and modelling each feature separately to identify normal or anomalous behaviour based on their individual statistical properties. This analysis typically involves computing summary statistics, such as mean, median, standard deviation, or quantiles, and examining the distribution of the variable using histograms, box plots, or probability density functions.

The univariate analysis enables the detection of outliers or deviations from expected behaviour by establishing thresholds or boundaries based on the statistical properties of individual variables. These thresholds can be set using techniques like z-scores, where observations that fall outside a certain number of standard deviations from the mean are flagged as anomalies. Additionally, univariate analysis can identify trends, seasonality, or patterns specific to a single variable.

However, it is important to note that univariate analysis alone may not capture complex relationships or interactions among variables, as it treats each variable independently. For a comprehensive understanding of the data and improved detection performance, multivariate analysis techniques, which consider the joint behaviour of multiple variables simultaneously, are often employed in statistical IDS [259].

a.2) Multivariate: In statistical Intrusion Detection Systems (IDS), the term ''multivariate'' refers to the analysis and modelling of multiple variables or features simultaneously. A multivariate approach considers the relationships and interactions among multiple variables to detect anomalies or intrusions in a system.

Traditionally, IDS systems have focused on analyzing individual variables independently, often using univariate methods. The univariate analysis treats each variable separately without considering the potential correlations or dependencies with other variables. However, in complex systems, such as network traffic or system logs, anomalies or intrusions may manifest as abnormal patterns across multiple variables rather than in isolation.

Multivariate statistical IDS takes a holistic approach by considering the joint behaviour of multiple variables. It explores the correlations, dependencies, and interactions among the variables to identify anomalous or suspicious patterns that may not be apparent when examining variables individually.

The multivariate analysis typically involves techniques such as:

1) Multivariate statistical models: These models, such as multivariate Gaussian distribution or multivariate time series models, capture the joint distribution of multiple variables. They estimate the system's expected behaviour and can identify deviations from this normal behaviour.
2) Dimensionality reduction: Multivariate IDS may employ dimensionality reduction techniques, such as Principal Component Analysis (PCA) or Singular Value Decomposition (SVD), to reduce the number of variables while preserving the most informative aspects

of the data. This can help simplify the analysis and improve efficiency.

3) Multivariate anomaly detection algorithms: These algorithms leverage statistical methods, machine learning, or data mining techniques to detect anomalies or intrusions based on patterns observed across multiple variables. They consider the relationships among variables and identify deviations from expected behaviour.

Multivariate statistical IDS can provide a more comprehensive and accurate assessment of system security by considering multiple variables simultaneously. It enables the detection of complex attacks or anomalies involving coordinated actions across multiple dimensions or variables. Additionally, multivariate analysis can help reduce false positives and enhance the overall performance of intrusion detection systems.

Ye et al. [259] developed a multivariate quality control technique to detect intrusions by creating a normal profile of regular activities over a long period of time. Due to the difficulty in estimating distributions for high-dimensional data, multivariate statistical IDs face their biggest hurdle.

a.3) Time Series Model: A time series refers to a set of data points collected over a defined time frame. If the probability of a new observation happening at that moment is unlikely, it is considered unusual [236]. In the context of statistical Intrusion Detection Systems (IDS), a time series model is a statistical modelling approach used to analyze and predict patterns in sequential data over time. Time series models are particularly relevant for IDSs as they can capture temporal dependencies and patterns in network traffic data, system logs, or other time-varying data sources.

Time series models aim to understand the underlying dynamics of a sequence of observations and make predictions based on historical patterns. They assume that the observed data points are not independent but depend on the previous observations sequentially. Time series models can detect anomalies or deviations from the expected behaviour by analysing historical data.

There are various types of time series models commonly used in statistical IDS:

1) Autoregressive (AR) models: AR models assume that the current value of a variable is a linear combination of its previous values, incorporating a weighted sum of lagged observations. These models are suitable for capturing short-term dependencies in the data.
2) Moving Average (MA) models: MA models, on the other hand, assume that the current value of a variable depends on the linear combination of the current and previous errors or residuals. MA models are effective in capturing sudden changes or shocks in the data.
3) Autoregressive Integrated Moving Average (ARIMA) models: ARIMA models combine AR and MA models and include differencing to handle non-stationary data. They are widely used for time series forecasting and anomaly detection in IDS.
4) Seasonal ARIMA (SARIMA) models: SARIMA models extend ARIMA models by incorporating seasonal components, making them suitable for capturing seasonal patterns in the data.
5) Exponential Smoothing models: Exponential smoothing models, such as Simple Exponential Smoothing (SES), Holt's Linear Exponential Smoothing, and Holt-Winters' Exponential Smoothing, are widely used for time series forecasting and detecting anomalies in IDS.

These time series models can be trained using historical data and then applied to predict future values or detect anomalies in real time. They provide a statistical framework to understand the behaviour of time-dependent data and make informed decisions regarding intrusion detection and network security.

It is important to note that the selection of an appropriate time series model depends on the characteristics of the data, the level of complexity desired, and the specific objectives of the IDS. A technique was suggested by Khraisat et al. [127] for identifying network irregularities by analyzing sudden changes in time series data.

b: KNOWLEDGE-BASED TECHNIQUES

The method in question is referred to as the expert system approach and requires constructing a comprehensive knowledge base that accurately represents the traffic pattern [91]. An intrusion is defined as any deviation from the expected traffic profile. The standard profile model is created using a set of rules based on human expertise to describe typical system behaviour. This model is different from other types of AIDS.

The advantage of this type of IDS is that it minimizes false-positive alerts as it is informed about all normal behaviours. Updating the knowledge base with the expected normal behaviour is a difficult and time-consuming task, especially in a computing environment that is constantly evolving. Obtaining information about all typical behaviours can be challenging.

The standard profile model is typically represented using states, transitions, and activities. For example, the model is capable of detecting changes in the input and making transitions based on the observed variations [241]. A finite state machine (FSM) is employed to monitor deviations from the expected behaviour, and any deviation from this FSM is considered an attack.

b.1) Finite State Machine (FSM): A Finite State Machine is a mathematical model that consists of a finite number of states and transitions between those states based on certain conditions or inputs. It is commonly represented as a directed graph, where nodes represent the states, and the transitions are represented by edges connecting the nodes.

In the context of AIDS, an FSM can be designed to represent a system or network's normal behavior or expected patterns. The FSM captures the sequence of events or states considered legitimate and indicates deviations from these expected patterns as potential anomalies.

The FSM-based approach involves the following steps:

1) Model Creation: The first step is to define the states, transitions, and conditions that constitute the system's expected behavior. This requires a thorough understanding of the system's expected behavior and the events or states that should occur in a specific order.
2) Training Phase: During training, the FSM is exposed to legitimate or normal data, allowing it to learn the patterns and transitions that define expected behavior. The FSM builds its knowledge base by capturing the sequences of states and transitions observed in the training data.
3) Anomaly Detection: Once the FSM has been trained, it can be used for anomaly detection. During the detection phase, incoming events or states are compared with the expected patterns the FSM defines. If a deviation or unexpected transition occurs, it is flagged as a potential anomaly or intrusion.
4) Alert Generation: When an anomaly is detected, an alert or notification can be generated to inform system administrators or security personnel about the potential intrusion or anomalous behavior. The alert can include details about the detected anomaly, such as the specific states or transitions that triggered it.

FSM-based techniques in AIDS provide a rule-based approach to anomaly detection, where the expected behavior is explicitly defined in the FSM model. By comparing observed events or states against this model, deviations from normal behavior can be identified and treated as potential security threats.

It's worth noting that while FSMs are effective for modeling certain types of systems, they may not capture complex or dynamic behaviors accurately. Therefore, combined with other techniques, such as statistical methods or machine learning algorithms, FSMs can be integrated into a more comprehensive IDS to enhance the accuracy and effectiveness of anomaly detection [171].

b.2) Description language: The description language provides a structured way to define and represent the features, behaviors, or attributes associated with known attacks or malicious activities. By utilizing a description language, security experts and system administrators can construct rules that capture the distinctive patterns and signatures of various attacks, allowing the intrusion detection system to identify and flag any deviations from normal behavior effectively.

Two examples of description languages mentioned are N-grammars and UML (Unified Modeling Language) [220]:

1) N-grammars: N-grammars are a type of formal language representation that define the syntax and structure of patterns or sequences in a system. N-gram-based description languages are commonly used for capturing sequential or temporal patterns in data, such as network traffic or system logs. By defining rules using N-gram-based description languages, specific attack patterns or sequences of events can be detected and classified as anomalies.
2) UML (Unified Modeling Language): UML is a standardized visual modeling language commonly used in software engineering and system design. While UML is primarily used for system modeling, it can also be leveraged as a description language in the context of intrusion detection systems. UML-based description languages provide a graphical notation for representing attack scenarios, system behaviors, and relationships between different components. By specifying rules using UML-based description languages, the system can identify instances where the observed behavior deviates from the expected system model, indicating a potential intrusion or anomaly.

In summary, a description language in the context of AIDS refers to a formal syntax used to write rules that describe the characteristics and patterns of specified attacks. These rules aid in detecting anomalies and intrusion attempts, allowing the system to identify and respond to potential security threats effectively.

b.3) Expert System: An expert system is a knowledge-based technique crucial in identifying and detecting attacks. An expert system consists of a set of rules that define various attack patterns or behaviors. These rules are typically created through a manual process by a knowledge engineer in collaboration with a domain expert who possesses deep knowledge and expertise in the field of network security.

The expert system's rules capture the knowledge and heuristics of the domain expert, reflecting their understanding of different types of attacks, their characteristics, and the indicators of compromise associated with them. These rules serve as a knowledge base that aids in identifying anomalous activities or behaviors that might indicate a potential intrusion.

During detection, the expert system analyzes incoming network data or logs and applies the defined rules to identify any matches or violations. When a match occurs, it signifies that the observed behavior aligns with a known attack pattern or malicious activity. This triggers an alert or warning to notify the system administrators or security personnel about the potential intrusion.

Expert systems are valuable in AIDS as they provide a mechanism to leverage expert knowledge and experience in the detection process. Encoding the domain expert's expertise into a set of rules enables the system to recognize known attack patterns and behaviors, even in complex and rapidly evolving network environments.

However, it's important to note that expert systems are typically limited to the knowledge and rules explicitly defined within them. They may struggle to detect novel or previously unknown attacks that fall outside the scope of the predefined rules. Therefore, expert systems are often complemented with techniques like machine learning or anomaly detection algorithms to handle unknown or emerging threats.

In summary, expert systems in the context of AIDS are knowledge-based techniques that utilize manually defined rules to identify and detect known attack patterns. They serve

as a valuable component in the overall intrusion detection system, leveraging the expertise of domain experts to enhance the system's ability to identify and respond to potential intrusions [128].

b.4) Signature Analysis: Signature analysis is a knowledge-based technique used for anomaly detection. It involves comparing incoming packets or network traffic against a predefined set of signatures to identify known patterns or malicious activities.

The initial approach in intrusion detection systems was string matching, which involved comparing each word or pattern in an incoming packet with a unique signature. These signatures are typically derived from known attack patterns or suspicious network behavior. When a match is identified between the packet content and a signature, an alert is generated to indicate a potential intrusion or anomaly. On the other hand, if no match is found, the traffic's metadata (such as source and destination addresses, ports, or protocols) is matched against the following signature in the signature database.

The signature database is a collection of predefined patterns, rules, or signatures representing known attack techniques or abnormal behavior. It is typically created and maintained based on prior knowledge and analysis of past attacks or malicious activities. The signatures may include specific strings, byte sequences, or behavior patterns associated with different types of attacks.

Signature analysis effectively detects well-known and documented attacks because it relies on matching against a predefined set of signatures. It allows for rapid detection and response to known threats. However, it has limitations when detecting novel or previously unseen attacks, as it heavily relies on the availability of signatures for known attacks. New or sophisticated attacks that do not match any existing signatures may go undetected.

In modern intrusion detection systems, signature analysis is often complemented with other techniques, such as anomaly detection and machine learning, to enhance the system's ability to detect known and unknown attacks. By combining multiple techniques, IDS can provide a more comprehensive and robust defense against various network intrusions and anomalies [119].

c: AIDS BASED ON MACHINE LEARNING TECHNIQUES

Machine learning involves extracting insights from large data sets. It involves using algorithms, methods, or complex mathematical functions that allow identifying and predicting patterns in the data and discovering significant information [73].

In the field of AIDS, machine learning is widely used to analyze data. Multiple techniques have been employed to uncover meaningful information from intrusion datasets, such as clustering, neural networks, association rules, decision trees, genetic algorithms, and nearest-neighbour methods [132], [253].

Chebrolu et al. investigated the effectiveness of two Bayesian networks (BN)-based feature selection techniques and combined these techniques for greater accuracy, including Classification Regression Trees (CRC) [62].

Thaseen and Kumar [229] proposed a random tree model for Network Intrusion Detection Systems (NIDS) to enhance accuracy and reduce the false alarm rate. Bajaj et al. proposed a feature selection approach that utilizes correlation-based attribute evaluation and information gain (IG) feature selection methods. The effectiveness of the selected features was evaluated using various classification techniques such as C4.5, naïve Bayes, NB-Tree, and Multi-Layer Perceptron [40], [126]. Decision tree methods were used to categorize the NSL-KDD dataset in order to build a model based on its metric data, and their performance was studied (see [222]). A combination of genetic algorithms and fuzzy rule mining was employed to evaluate the importance of features in IDS [75].

The goal of utilizing machine learning is to develop intrusion detection systems (IDS) that are more precise and rely less on human expertise in identifying patterns. In total, there are three main types of machine learning methods, which include supervised, unsupervised, and reinforcement learning. These methods aim to help build IDS based on the given data set.

d: SUPERVISED LEARNING IN INTRUSION DETECTION SYSTEM

This section provides a detailed overview of various supervised learning techniques for intrusion detection systems (IDS).

Supervised learning-based IDS methods identify intrusions by utilizing labelled training data. The process is divided into two phases, training, and testing. During the training phase, relevant features and classes are identified, and the algorithm learns from the data samples. Each record consists of a network or host data source and its corresponding label, either intrusion or normal. Feature selection is then performed to eliminate irrelevant features, and a classifier is trained using a supervised learning technique, using the selected features to determine the relationship between the input data and the output label. The trained model is then used to classify new, unseen data as either intrusion or normal during the testing phase.

A general method for using classification algorithms is shown in Figure 5. A number of measures are discussed in Section V for measuring a classifier's performance in terms of its capacity to predict the proper class.

The following paragraphs discuss various classification techniques, including decision trees, rule-based systems, neural networks, support vector machines, naïve Bayes, and nearest-neighbour methods.

d.1) Decision Trees: Decision trees are a popular and effective approach for classification. A decision tree is a flowchart-like structure that uses a tree-like model of decisions and

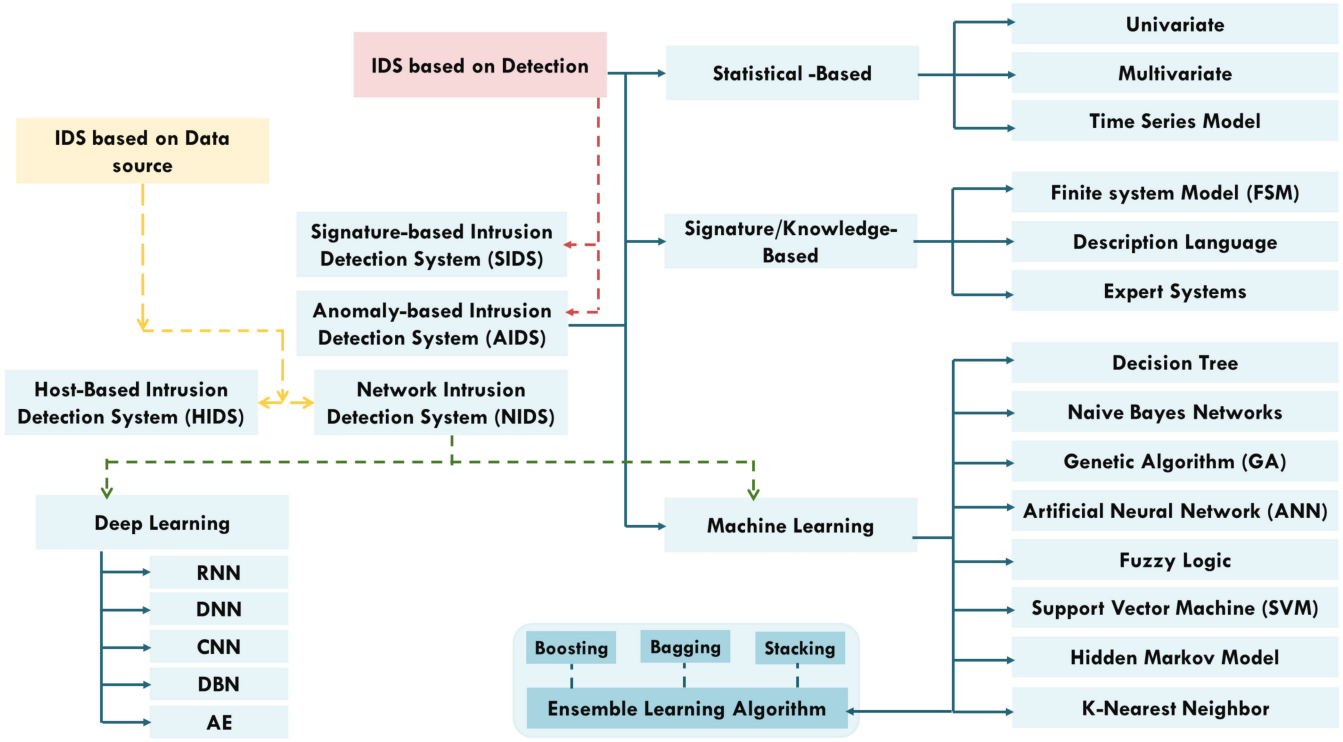

FIGURE 4. Typical ML & DL based NIDS methodology.

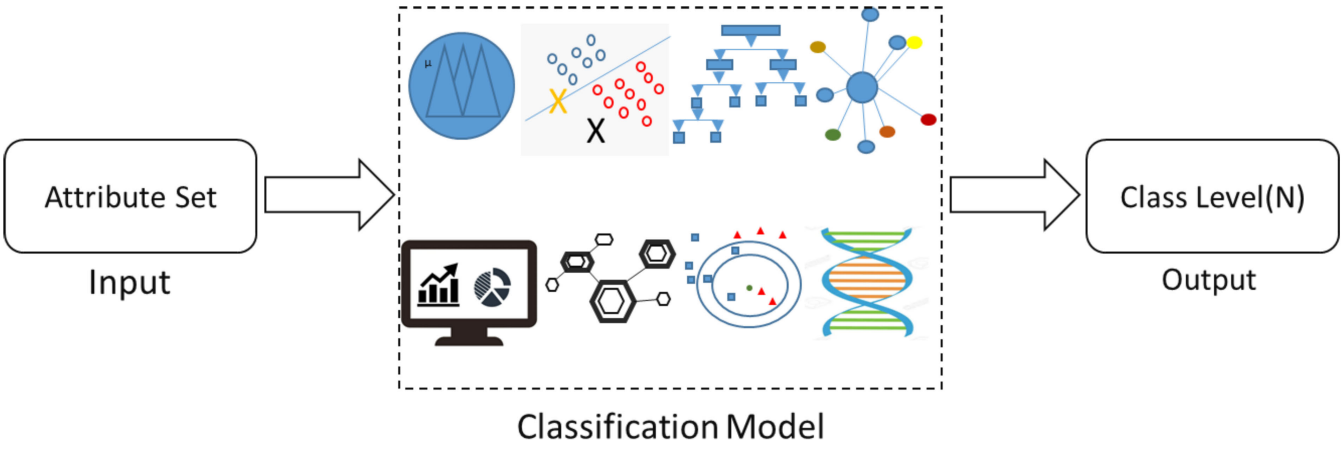

FIGURE 5. Classification methods for label data.

their possible consequences. It is built based on the available training data, where each internal node represents a test on a specific attribute, each branch represents the outcome of the test, and each leaf node represents a class label or a decision.

The decision tree algorithm aims to create an optimal tree to classify instances based on their attribute values efficiently. The tree is constructed recursively, starting from a root node and splitting the data at each internal node based on the attribute that provides the best discriminatory power. This process continues until the algorithm determines that a particular branch reaches a leaf node, which corresponds to a specific class label or decision.

Decision trees in IDS follow a similar structure. The decision node serves as the initial element and specifies a test attribute related to the intrusion detection process. This attribute could be based on features such as network traffic characteristics, system logs, or behavior patterns. The branches emerging from the decision node represent potential courses of action depending on the test attribute's result.

Each branch corresponds to a specific outcome or value of the tested attribute.

At the leaf nodes, decisions are made about the class label or category to which the instance belongs, such as normal or malicious. These leaf nodes serve as the final classification outcomes of the decision tree [201].

Several prominent decision tree algorithms have been developed and utilized in the field of IDS. The C4.5 algorithm [193], the ID3 algorithm [192], and the CART algorithm [53] are among the well-known decision tree algorithms used for intrusion detection. These algorithms employ various techniques for attribute selection, pruning, and handling missing values to construct accurate and robust decision trees for classifying network traffic or system events as usual or intrusive.

Overall, decision trees offer a transparent and interpretable approach to intrusion detection, allowing analysts to understand the decision-making process and identify important features contributing to the classification. They can effectively handle complex and nonlinear relationships between attributes and provide valuable insights for detecting and mitigating security threats in network and system environments.

d.2) Naive Bayes: The Naive Bayes approach calculates the probability of a certain type of attack based on observed events using the conditional probability formula. It is based on features with varying probabilities of occurring in attacks and normal behaviour. It is a popular intrusion detection system (IDS) model because of its ease of use. Murray et al. [168] used genetic algorithms (GA) to develop simple rules for network traffic. A genome represents each rule, and the initial population of genomes comprises several random rules. Each genome consists of different genes that correspond to characteristics such as source IP, destination IP, source port, destination port, and protocol type [100].

d.3) Artificial Neural Network (ANN): The use of ANN has become popular for detecting various forms of malware due to its effectiveness. The most commonly used supervised learning technique is the backpropagation (BP) algorithm. It assesses the gradient of the network's error with regard to its adjustable weights.

However, there is room for improvement in detection precision and accuracy for ANN-based IDS, particularly for less common attacks. It is challenging for the ANN to accurately understand the attributes of these attacks since the training dataset for less frequent attacks is less than that for more frequent attacks. As a result, for less frequent attacks, detection accuracy is lower. If low-frequency assaults are not discovered in the field of information security, severe harm may result. Suppose that User to Root (U2R) attacks are not detected, then an attacker could gain the authorization capabilities of a root user and perform malicious activities on the victim's computer systems. Additionally, the rarer attacks frequently represent outliers [244]. Learning can take a very long time with ANNs since local minima are a common problem. When combined with one or more hidden layers, ANN can create highly nonlinear models that capture complex relationships between input variables and classification outcomes, which is considered an advantage of this machine-learning technique.

d.4) Fuzzy Logic: Instead of the standard true or false Boolean logic that modern PCs are built on, this method is based on degrees of uncertainty. As a result, it offers a simple method for drawing a final judgment from input data that is murky, confusing, noisy, erroneous, or lacking.

Fuzzy logic allows an instance to simultaneously belong, potentially partially, to numerous classes in a fuzzy domain. The acquired data for the intrusion detection problem also includes a number of derived statistical metrics and various numerical properties. IDSs that use inflexible criteria and depend on quantitative data often produce numerous false alerts. Even a minor departure from a pattern may go unnoticed, and insignificant variations in typical behaviour may trigger false alarms. To keep the false rate low, it is possible to model this tiny aberration using fuzzy logic. Elhag et al. showed that the rate of false alarms in identifying invasive acts might be reduced by using fuzzy logic. They described a set of fuzzy rules to distinguish between normal and aberrant computer system behaviour, as well as a fuzzy inference engine to identify intrusions [75].

d.5) Support Vector Machines (SVM): Support Vector Machines (SVMs) are a type of discriminative classifier that uses a separating hyperplane to categorize intrusions. To linearly classify intrusions, SVMs utilize a kernel function that maps the training data into a higher dimensional space. SVMs are known for their strong generalization ability and are particularly useful when working with many attributes and limited data points. Different types of separating hyperplanes can be generated by applying various kernels, such as linear, polynomial, Gaussian Radial Basis Function (RBF), or hyperbolic tangent. Many features in the IDS dataset are redundant or have less impact on classifying data items into the appropriate groups. Therefore, when training an SVM, feature selection should be considered. Multiple class classification is another application for SVM. A method was proposed using an SVM classifier with a radial basis function (RBF) kernel to categorize the KDD 1999 dataset into predetermined classes [138].

d.6) Hidden Markov Model (HMM): A Hidden Markov Model, a statistical Markov model, assumes that the system being analyzed is a Markov process with unknown information. HMMs can be utilized to categorize specific types of malware [170]. An HMM is trained based on identified characteristics of malware. This trained model is then utilized to assess incoming network traffic. The evaluation produces a score that is compared to a predetermined threshold. If the score surpasses the threshold, the traffic is classified as malicious, whereas if the score falls below the threshold, it is considered normal.

d.7) K-Nearest Neighbours (KNN) Classifier: The k-Nearest Neighbor (k-NN) algorithm is a commonly used

classification technique in machine learning that does not require a predetermined model. Its goal is to assign a category to an unlabeled data point by analyzing the class of its k closest neighbouring data points. The value of k is a predetermined integer that determines the number of nearby data points to consider. Figure 6 shows a KNN classifier where k = 5. Here, point X represents an instance of unlabeled data that needs to be classified. There are three identical patterns from the class Intrusion and two from the class Normal among X's five closest neighbours. The decision to assign X to the Intrusion class can be made with a majority of votes. The k-NN algorithm is often considered a benchmark for other classifiers due to its superior classification results in many intrusion detection systems [143].

e: UNSUPERVISED LEARNING IN INTRUSION DETECTION SYSTEM

Unsupervised learning is a machine learning technique that is applied to datasets without pre-existing class labels in order to uncover patterns and relationships within the data. Instead of relying on labelled data to train a model to make predictions, unsupervised learning treats the input data as a set of random variables and constructs a joint density model to describe the dataset. Through the process of unsupervised learning, the data is categorized into different groups or classes without the use of any pre-existing labels or training data. This stands in contrast to supervised learning, which relies on labelled data to guide the learning process. In the context of creating an intrusion detection system, unsupervised learning means using a method to detect intrusions by training the model with unlabeled data.

As shown in Figure 7, In the clustering process of records, anomalies are labelled as intrusions because they appear in small clusters, while normal occurrences tend to form larger clusters. Normal and intrusion instances are different, so they do not fall into an identical cluster.

e.1) K-Means: The K-means algorithm is a popular clustering technique used to group a set of 'n' data points into 'k' clusters. The algorithm assigns each data point to the cluster whose mean is closest to that point. It is a distance-based approach that uses the Euclidean distance measure to determine the similarity between data points. Unlike other clustering techniques, the K-means algorithm does not require computing the distances between all possible pairs of data points, which makes it computationally efficient. The user predetermines the number of clusters, and multiple solutions may be tested before selecting the best one. One method Utilized the K-means clustering algorithm to distinguish various profiles of host behaviour [170]. Researchers have proposed new distance measurement techniques to enhance the performance of the k-means clustering algorithm for intrusion detection. These techniques have shown promising results, leading to better outcomes using this unsupervised method. The findings show that k-means is a more effective approach for classifying data when multiple types of datasets are available. Clustering can be applied in intrusion detection systems to streamline intrusion signatures, produce a more accurate signature, or group similar intrusions.

e.2) Hierarchical Clustering: Hierarchical clustering is a clustering technique that aims to create a hierarchy of clusters by recursively dividing or merging data points based on their similarity or dissimilarity. It organizes the data points in a tree-like structure known as a dendrogram, which illustrates the relationships between clusters at different levels of granularity.

There are two main types of hierarchical clustering:

(i) Agglomerative- This approach starts with each data point as a separate cluster and iteratively merges the most similar clusters until a single cluster containing all the data points is formed. The algorithm combines the closest clusters at each step based on a defined distance metric, such as Euclidean or Manhattan distance.

(ii) Divisive - This approach takes the opposite approach of agglomerative clustering. It begins with a single cluster containing all the data points and recursively splits it into smaller clusters until each data point forms its own individual cluster. At each step, the algorithm identifies the cluster with the highest dissimilarity and divides it into two clusters.

The choice of distance metric and linkage criterion is crucial in hierarchical clustering. The linkage criterion determines how the distance between clusters is measured and affects the resulting cluster structure. Some common linkage criteria include:

1) Single linkage: The distance between two clusters is defined by the shortest distance between any two points in the two clusters.
2) Complete linkage: The distance between two clusters is defined by the maximum distance between any two points in the two clusters.
3) Average linkage: The distance between two clusters is defined by the average distance between all pairs of points from the two clusters.

Hierarchical clustering provides a flexible approach to clustering as it can handle different shapes and sizes of clusters. It also visually represents the cluster hierarchy through dendrograms, allowing users to explore and interpret the relationships between clusters. However, hierarchical clustering can be computationally expensive, especially for large datasets, and its performance heavily depends on the choice of distance metric and linkage criterion [169], [175].

e.3) Probabilistic Clustering: Probabilistic clustering is a clustering technique that incorporates probabilistic models to assign data points to clusters. Unlike traditional clustering algorithms that assign data points to clusters based on distance or similarity measures, probabilistic clustering considers the uncertainty associated with each assignment. It assumes that each data point belongs to a particular cluster with a certain probability.

One popular example of probabilistic clustering is the Gaussian Mixture Model (GMM). In GMM, each cluster is represented by a multivariate Gaussian distribution characterized by its mean and covariance matrix. GMM aims to

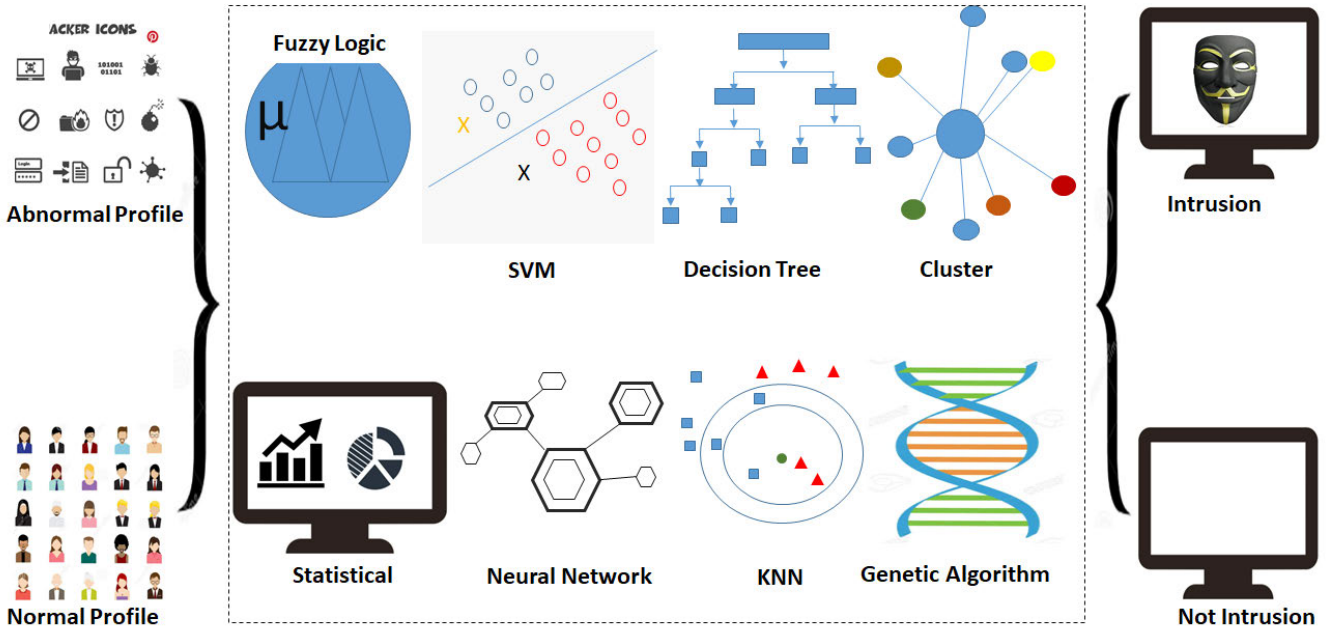

FIGURE 6. Conceptual working of AIDS approaches based on machine learning.

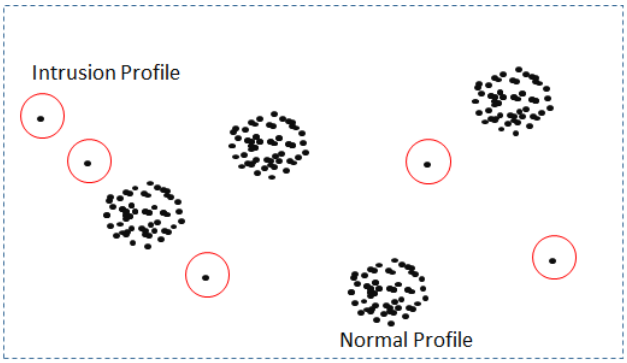

FIGURE 7. Using clustering for intrusion detection.

estimate the parameters of these Gaussian distributions and assign data points to clusters based on the probability of belonging to each distribution.

The probabilistic nature of clustering algorithms allows for more flexibility and provides a richer representation of the underlying data structure. It enables the identification of overlapping or fuzzy boundaries between clusters, where data points may have varying degrees of association with multiple clusters. This is in contrast to deterministic clustering algorithms, such as k-means, which assign data points to a single cluster without considering the uncertainty in the assignment.

Probabilistic clustering techniques find applications in various fields, including pattern recognition, image segmentation, data mining, and bioinformatics. They offer a probabilistic framework to model complex data distributions, capture hidden structures, and handle data points that do not clearly belong to a single cluster [43], [58].

e.4) Principal Component Analysis: Principal Component Analysis (PCA) is a widely used technique for dimensionality reduction and feature extraction. It aims to identify a subset of low-dimensional features from a larger set of features while retaining as much of the original information as possible.

PCA achieves this by transforming the original features into a new set of uncorrelated variables called principal components. These principal components are ordered so that the first component captures the maximum variance in the data, the second component captures the second-highest variance, and so on. By selecting a subset of the top-ranked principal components, one can effectively reduce the dimensionality of the data while preserving the most important information.

The process of performing PCA involves the following steps:

1) Standardization: If the original features have different scales, it is important to standardize them to have zero mean and unit variance. This ensures that features with larger scales do not dominate the PCA process.
2) Covariance Matrix Calculation: The covariance matrix is computed based on standardized features. It represents the relationships and dependencies between the features.
3) Eigenvector-Eigenvalue Decomposition: The covariance matrix is then decomposed into its eigenvectors and eigenvalues. The eigenvectors represent the directions or axes of maximum variance in the data, while the eigenvalues indicate the variance each eigenvector explains.
4) Selection of Principal Components: The eigenvectors are ranked based on their corresponding eigenvalues,

and the top-ranked eigenvectors are chosen as the principal components. The desired level of dimensionality reduction determines the number of principal components selected.

PCA has several applications, including data visualization, noise reduction, and feature selection. Reducing the dimensionality of the data can simplify subsequent analysis tasks, improve computational efficiency, and help mitigate the curse of dimensionality. However, it is important to note that PCA may not always be suitable for all datasets or analysis goals, and its effectiveness depends on the underlying structure of the data [17], [54].

e.5) Singular Value Decomposition: Singular Value Decomposition (SVD) is a matrix factorization technique that breaks down a matrix into three separate matrices, providing a linear approximation of the original matrix. It is widely used in various fields, including data analysis, signal processing, and machine learning, for dimensionality reduction, data compression, and feature extraction.

The goal of SVD is to find a set of features that captures the underlying structure and meaning of the data. It decomposes an original, typically rectangular, matrix into three matrices: U, Σ, V.

1) U represents the left singular vectors and describes the relationship between the rows of the original matrix.
2) Σ is a diagonal matrix that contains the singular values, which represent the importance of each feature or dimension. The singular values are sorted in descending order, with the highest singular value corresponding to the most significant feature.
3) V represents the right singular vectors and describes the relationship between the columns of the original matrix.

One can approximate the original matrix while retaining the most important information by selecting a subset of the highest-ranked singular values and their corresponding singular vectors. This allows for dimensionality reduction and feature extraction, as the selected singular vectors are the reduced features that best represent the original data.

The matrix approximation achieved through SVD can be used for various purposes, such as data compression, denoising, and identifying dominant patterns or structures within the data. It is particularly useful when dealing with high-dimensional or noisy data, as it provides a compact representation of the data while preserving the most relevant information.

In the context mentioned, Singular Value Decomposition can help identify the best set of features for anticipating detection by selecting the most important singular vectors that capture the meaningful structure of the data [41].

e.6) Independent Component Analysis: It is used for illuminating hidden aspects that support collections of random characteristics.

In Cyber-Physical Control Systems (CPCS) field, many studies have been conducted using unsupervised learning for attack detection and reactive mitigation, including the proposal of redundancy-based resilience methods [31]. Chao Shen and his team developed a specialized network sublayer that collects context information from driver nodes within the control network and uses data mining methods, including k-means and k-nearest neighbour, to distinguish between different views. They also introduced the Hybrid-Augmented device, which is designed to detect intrusions in industrial control system (ICS) networks by analyzing network packets using machine learning techniques and filtering out anomalous traffic [212].

f: SEMI-SUPERVISED LEARNING

Semi-supervised learning is a technique that lies between supervised learning, which uses fully labelled training data, and unsupervised learning, which has no categorized training data. Studies have demonstrated that using semi-supervised learning in intrusion detection systems can improve classifier performance, requiring less effort and cost. It is an effective solution for many IDS problems where labelled data may be scarce or limited [38].

Some semi-supervised learning are, boosting based semi-supervised learning methods [49], [262], Semi-Supervised SVM, graph-based methods [202], Expectation Maximization algorithms [83], co-training [195] and self-training [49].

g: HYBRID-BASED TECHNIQUES

Traditional IDSs have drawbacks such as difficulty in modification, inability to recognize new malicious threats, poor accuracy, and a high rate of false alerts. Where AIDS has a drawback, like a high percentage of false positives. Hybrid IDS combine AIDS and SIDS. It addresses the drawbacks of both SIDS and AIDS. Farid et al. proposed a hybrid intrusion detection system that uses a combination of Naive Bayes and decision tree algorithms, and it results in the detection rate of 99.63 % on the KDD'99 dataset [77].

h: ENSEMBLE METHODS

Multiple machine learning algorithms might be employed to achieve greater prediction performance than any of the individual learning algorithms. Various ensemble techniques have been put forward, including Bagging, Boosting, and Stacking.

To put it differently, boosting refers to a set of algorithms that can improve the performance of weak models by transforming them into stronger ones. Bagging involves training a classifier using multiple subsets of the same data. Stacking involves merging the predictions of multiple classifiers through the use of a meta-classifier [18]. In order to train the meta-model, the outputs of the base models are used as inputs after the base models have been generated by utilizing the entire training set.

Jabbar and his team proposed an ensemble classifier that combines Random Forest and Average One-Dependence Estimator (AODE) algorithms to tackle the problem of attribute dependence in Naive Bayes. The use of Random

Forest (RF) improves accuracy and decreases false positive results [109]. Merging both methods in an ensemble leads to increased accuracy compared to using either approach alone.

A novel fuzzy semi-supervised learning technique was introduced by Rana and co-authors, which leverages both labelled and unlabeled samples, as well as a supervised learning algorithm, to improve the effectiveness of classifiers for Intrusion Detection Systems (IDSs). A Single Hidden Layer Feed-Forward Neural Network (SLFN) is trained to generate a fuzzy membership vector, which is then utilized to categorize the unlabeled sample data into low, medium, and high levels of fuzziness [38]. Experimental results indicate that the unlabeled samples from the low and high fuzziness groups have the greatest impact on improving the accuracy of the IDS compared to traditional methods.

i: REINFORCEMENT LEARNING

Integrating deep learning and reinforcement learning is employed in Deep Reinforcement Learning to develop Intrusion Detection Systems (IDSs), which involve an agent interacting with its environment. The agent must learn to interact with its environment to achieve its goals.

Deep reinforcement learning is the application of reinforcement learning to train deep neural networks, which comprise an input layer, an output layer, and multiple hidden layers, similar to conventional deep neural networks. For example, consider a scenario where a bus is trying to transport its passengers. The output is a collection of alternative actions, such as speeding up, slowing down, turning left, or turning right. Our inputs are position, speed, and direction. In order to learn what activities result in favourable outcomes given a particular condition of the environment, we are also putting our signal into the network.

i.1) Deep Q-network Deep neural networks and reinforcement learning are coupled at scale. The algorithm was created using deep neural networks to improve the Q-Learning conventional RL technique.

i.2) Double Q-learning Double estimation is used in this off-policy reinforcement learning approach to address overestimation issues with conventional Q-learning.

V. PERFORMANCE METRICS FOR IDS

The performance of an Intrusion Detection System is assessed using standard evaluation measures:

True Positive Rate (TPR): It refers to the proportion of correctly predicted attacks out of the total number of attacks. If every intrusion is found, the TPR is 1, which is incredibly uncommon for an IDS. The Detection Rate (DR) or the Sensitivity is the other name for TPR. TPR expressed mathematically as

$$TPR = \frac{TP}{TP + FN}$$

False Positive Rate (FPR): It represents the ratio of normal instances that are incorrectly labelled as an attack over the total number of normal instances.

$$FPR = \frac{FP}{FP + TN}$$

False Negative Rate (FNR): Measure of how many anomalous instances are incorrectly classified as normal by the IDS. It is calculated as the ratio of the number of missed anomalous instances to the total number of actual anomalous instances.

$$FNR = \frac{FN}{FN + TP}$$

Classification Rate (CR) or Accuracy: This metric evaluates the precision of the Intrusion Detection System (IDS) in detecting normal or abnormal traffic patterns, and is represented as the proportion of accurately predicted instances out of all instances.

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

Precision: It is referred to as the ratio of correctly predicted Attacks to all the samples predicted as Attacks.

$$FNR = \frac{TP}{TP + FP}$$

F-Measure: The harmonic mean of the Precision and Recall is how it is defined. In other words, it is a statistical technique for evaluating a system's accuracy while taking into account both its precision and recall.

$$FMeasure = 2\left(\frac{Precision \times Recall}{Precision + Recall}\right)$$

Training Time (T1): It describes the time required for an approach to train the whole dataset and to build the NIDS model with the best fit as in:

$$T1 = end^{Training\ Time} - start^{Training\ Time}$$

Testing Time (T2): It Describes the time required for as approach to predict the whole dataset as either normal or attack as in

$$T1 = end^{Testing\ Time} - star^{Testing\ Time}$$

VI. INTRUSION DETECTION DATASETS

The evaluation datasets are essential to the validation of any IDS approach because they let us gauge how well the suggested method can identify intrusive activity. There are publicly available datasets that are widely used as benchmarks. This section discusses the features and limitations of the existing datasets that are used to build and evaluate IDS in comparison.

1) DARPA/KDD CUP99

The KDD98 dataset, produced by MIT Lincoln Labs for DARPA in 1998, is considered the first dataset for Intrusion Detection Systems (IDS) created to provide a comprehensive and accurate benchmarking environment. Despite its widespread use, it has limitations in terms of its reflection on real-world scenarios [68].

TABLE 4. Merit and demerits of ML/DL-based techniques.

Article	Advantages	Disadvantages	Cause
Yin et al. [263]	NIDS-based RNN model shows high DR using ML	Complex model and delay in training. Used old dataset NSLKDD and low DR for attacks R2L, U2R.	Smaller RNN model
Shen et al. [212]	Combination of BAT optimization algorithm and Ensemble method based on ELM as base classifier way outperforms individual ELM performance.	Used old datasets KDD Cup'99, NSL-KDD, Kyoto and low DR for attacks like U2R.	Limited processing power and a fixed CPU load
Shone et al. [215]	A non-symmetric deep Autoencoder with NIDS was used to decrease complexity and improve the efficiency of feature selection, in combination with a Random Forest classifier.	Used old datasets KDD Cup'99, NSL-KDD and low performance for attacks like R2L, U2R.	Insufficient training data
Ali et al. [108]	Used NIDS based FLN and particle swarm optimization algorithm showed better performance than other similar algorithms.	Used old dataset KDD Cup'99 and low performance for attacks like R2L, U2R.	Limited training data
Jia et al. [111]	In comparison to ML-based IDS, a NIDS that employs a DNN with four hidden layers shows superior performance.	Used old datasets KDD cup'99, NSLKDD and low DR for attacks like U2R.	Randomly selected training sets (90%)
Wang et al. [221]	The impact of a DNN-based IDS against adversaries was investigated, and the effectiveness of current attack algorithms was assessed against the DNN-based IDS.	Used old dataset NSLKDD.	Old dataset
Yan et al. [258]	An effective NIDS is proposed by utilizing a Stacked Sparse Autoencoder (SSAE) for feature extraction in conjunction with an SVM as the classifier.	Used old dataset NSL-KDD and reasonable DR comparing the other still lower for U2R and R2L.	Scarcity of R2L and U2R attack samples
Naseer et al. [173]	A comparison between ML and DL-based NIDS is conducted by implementing both algorithms on a testbed integrated with GPU.	Used old dataset NSL-KDD for evaluation.	Old dataset
Xu et al. [256]	A solution for NIDS is proposed by utilizing GRU as the main memory for RNN, in combination with Multilayer Perceptron and Softmax classifier.	Used old dataset KDD Cup'99 and NSL-KDD and low performance for attacks like R2L, U2R.	Insufficient number of R2L and U2R attacks
Al-Qatf et al. [28]	A NIDS is developed using a self-taught learning approach using Sparse AE and SVM.	Used old dataset NSL-KDD and no performance evaluation was shown for attacks like R2L and U2R.	Old dataset
Marir et al. [153]	A method for detecting abnormal behaviour in a distributed manner is proposed by utilizing a DBN for feature extraction, an ensemble of SVM for detection, and a voting mechanism for prediction.	Complex model and delay in training for slightly deeper layers.	Higher complexity
Papamarti et al. [183]	An Auto Misuse Detection System is developed by combining Self-taught Learning, MAPE-K Frameworks, and Sparse Autoencoder for learning significant features.	Used old datasets KDD Cup'99 and NSLKDD and low DR for attacks like R2L, U2R.	State's high deviation from initial training set
Khan et al. [121]	A two-stage NIDS using Autoencoder is proposed. Improved DR for minority attack classes is achieved by preprocessing techniques like downsampling and SMOTE.	DR for KDD Cup'99-99.99%, UNSW-NB15-89.13%, 10% lower for newer Dataset, and no explanation about minority attack class.	Not available
Xiao et al. [255]	NIDS-based CNN DL algorithm. Dimensionality reduction, FE performed using PCA and AE.	Performance evaluated on KDD Cup'99 dataset. Low detection rate for U2R and R2L classes.	Flawed policy
Yao et al. [260]	Clustering combine with RF used in multilevel IDS. Shows superior DR for the attacks on the dataset with low instances.	Old dataset KDD Cup'99 used for model validation.	Old dataset
Vinayakumar et al. [239]	The Scale-Hybrid-ID-AlertNet; DNN-based real-time monitoring used various datasets.	Complex model; lower DR for minority attack class.	Additional features required
Gao et al. [81]	Adaptive Ensemble Model using base classifiers such as DT, RF, K-NN, and DNN. The adaptive voting algorithm is used to select the best classifier.	Model tested on NSL-KDD dataset, unsatisfactory results for weaker attack classes.	Imbalanced sample proportions
Wei et al. [249]	Optimization of DL Based DBN using Particle Swarm, Fish Swarm, and Genetic Algorithm.	Model complex requires more training time; evaluated on NSL-KDD dataset.	
Zhang et al. [267]	Multilayer NIDS model based on CNN & gcForest; P-Zigzag algorithm introduced to convert raw data into a 2D grayscale image; tested by combination UNSW-NB15 & CIC-ID2017 datasets.	DR for attack classes with less training data is low.	Extremely imbalanced amount of data
Malaiya et al. [151]	Several IDS models are developed using FCN, VAE, and Seq2Seq structures. The models are tested on both new and older datasets.	Seq2Seq model performs better than others with higher training costs. Unclear about best for detecting minority attack classes.	Higher training costs (Seq2Seq)
Karatas et al. [117]	Six ML-based NIDS models analyzed to tackle dataset imbalance through SMOTE, resulting in improved DR for minority attack class; tested using new CSECIC-ID2018 dataset.	Adaboost algorithm achieves higher DA at the cost of higher execution time.	Sequential training process

TABLE 4. *(Continued.)* **Merit and demerits of ML/DL-based techniques.**

Jiang et al. [112]	Deep hierarchy IDS is proposed using CNN and BiLSTM. Class imbalance problem solved by SMOTE. Tested by both old (NSL-KDD) & and new (UNSW-NB15) datasets.	Model complex, detection of minority data classes slightly improved, low compared to other attack classes.	Very limited number of samples in training sets
Yang et al. [259]	Proposed NIDS uses adversarial variational AE with regularization and DNN. Tested on NSL-KDD and UNSW-NB15 datasets.	Performance on the NSL-KDD dataset shows reasonable DR for U2R and R2L attacks, lower for other attacks. DR for minority class attacks in UNSW-NB15 not disclosed.	Imbalanced data
Yu et al. [256]	Efficient NIDS model proposed using few-shot learning in DL. DNN/CNN is used for embedding to reduce dimension and extract features. Evaluated on NSL-KDD and UNSW-NB15 datasets.	Good in detecting U2R & R2L for NSL-KDD; lower for other attack classes; Limitations (FSL) due to the need for labelled data, restricting frequent training with unlabeled data for better detection.	Need for labelled data
Andresini et al. [37]	Multi-stage intrusion detection proposed using AE with a convolution layer and two stacked fully connected layers. Tested on new and older datasets.	Model's effectiveness in detecting minority class not specified. No details on attack structure and characteristics were provided.	Not available
J. Gao et al., Sarkar et al. [80] [206]	Automatically extract and select features without requiring an extensive pre-processing procedure.	Although the CCRBM method has shown promising results, additional research is necessary to discover a more efficient approach for scene recognition and feature extraction.	Inherent complexity and design choices
Bhattacharya, Lane [47]	Utilize sparsification approach through sparse coding to separate convolutional kernel and fully connected layer, which helps in reducing computation time and memory usage.	Often, the inference is performed over the collected sensor data.	Not available
Khan, Taati [123]	Automatically extract generic features from unprocessed sensor data.	Addressed rare event (fall) classification model using AE, but parameter tunings such as learning rate, network topology, and activation functions required for better function.	Poor parameter tunings
Ijjina , Mohan [104]	Improve performance generalization and achieve high model diversity.	To accomplish this, one can manipulate the input features and initialize the models in a certain way.	Input feature manipulation
Lin et al., Ravi et al. [142] [196] [197]	Improve human activity recognition in real-time onboard with reduced feature vectors, optimal decomposition of complex activity details and enhanced generalization ability of DL algorithms.	Limitations in the scalability of model inference. Showed computation time obtained from low-power devices.	Scalability
Alzantot et al. [34]	Differentiate genuine from artificially generated data sets to enhance data privacy during collection.	Discriminating artificial sensory data in human activity recognition.	Not available

2) CAIDA

The CAIDA dataset, compiled in 2007, contains network traffic traces related to Distributed Denial-of-Service (DDoS) attacks [182]. However, it has the disadvantage of limited diversity in the types of attacks it covers. Additionally, the data collected does not capture characteristics from the entire network, making it difficult to differentiate between normal and abnormal traffic patterns.

3) NSL-KDD

The NSL-KDD dataset was developed from the KDD cup99 dataset [226] and made available to the public. A statistical analysis of the cup99 dataset revealed significant issues that negatively impacted the accuracy of intrusion detection and resulted in an incorrect evaluation of the system.

4) ISCX 2012

This dataset, as described in a study by Shiravi et al. [214], analyzed real network traffic traces from protocols such as HTTP, SMTP, SSH, IMAP, POP3, and FTP to identify typical computer behaviour. The dataset is based on labelled, realistic network traffic that encompasses a range of attack scenarios.

5) ADFA-LD AND ADFA-WD

The ADFA-LD and ADFA-WD datasets were developed by researchers at the Australian Defence Force Academy and made available as open-source datasets to demonstrate the current attack structures and techniques [67]. These datasets were created to evaluate system-call-based HID (Host-based Intrusion Detection). Linux ubuntu v11.04 is used as OS to build ADFA-LD [68]. The ADFA-LD dataset is appropriate for illustrating the contrasts between SID and AIDS methods for intrusion detection because it includes attack instances that originated from new zero-day malware. It consists of three different types of data, all of which consist of raw system call records. The training datasets were collected from the host computer during normal user activities, such as browsing the web and creating LATEX documents.

6) CICIDS 2017

The dataset includes normal behaviour and information on new malware attacks, including Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet, and DDoS [210]. The dataset labelling includes a timestamp, source and destination IP addresses, source and destination ports, protocols, and type of attack. The data was collected

using a complete network setup that encompasses nodes operating Linux, macOS iOS, various versions of Microsoft Windows (including Windows 10, Windows 8, Windows 7, and Windows XP), as well as modems, firewalls, switches, and routers.

7) KYOTO 2006+

The Kyoto University created this dataset using network traffic records obtained from honeypots, darknet sensors, email servers, web crawlers, and other network security tools [218]. The most recent dataset version includes traffic records from 2006 to 2015, with each record containing 24 statistical features. Fourteen of these features are from the KDD Cup'99 dataset, and the remaining ten are additional features.

8) CSE-CIC-IDS2018

The Communications Security Establishment (CSE) and CIC collaborated in 2018 to develop this dataset [117]. It was created by forming user profiles that encapsulate a summarized version of different events and combining these profiles with a unique set of features. The dataset includes seven attack scenarios: Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and network infiltration from within.

A. PUBLIC IDS DATASET COMPARISON

Since machine learning techniques are utilized to manage AIDS, the datasets utilized to assess these techniques are critical for a practical evaluation. Table 5 summarises datasets along with analysis techniques and results for each dataset from prior research.

VII. IDS FEATURE SELECTION

Feature selection aids in reducing computational complexity, removing redundant data, enhancing the accuracy of machine learning methods, simplifying data, and decreasing false alarm rates. Several methods have been employed in this field of research to develop lightweight intrusion detection systems (IDSs).

Data is extensively generated across different fields like social media, healthcare, network security, and education, leading to a significant challenge known as the curse of dimensionality. This problem refers to the sparsity of data when it is transformed into a high-dimensional space. Additionally, approaches designed to handle datasets with numerous features struggle to achieve satisfactory performance as they tend to overly tailor the data that is not yet known. Analyzing large datasets demands increased memory capacity and computational resources due to their size [61]. In this context, feature engineering has emerged as a valuable solution for managing high-dimensional data. Feature engineering is a highly productive and extensively studied field in various domains, such as pattern recognition [162], data mining (Talavera 2005) [272], machine learning [122]. For other applications, it is being used such as text categorization [176], intrusion detection [243], and image retrieval [263].

Feature engineering has emerged as a powerful and logical approach for effectively managing both low and high-dimensional data in order to tackle classification problems. Through extensive empirical analysis, feature engineering has incorporated simpler and more comprehensive models, thereby improving the performance of techniques and enabling the creation of refined and comprehensive data. The current surge in data poses significant challenges in data handling, leading to increased reliance on feature engineering. In this section, we present a substantial understanding of feature engineering research, which is motivated by various data-related issues such as redundant and irrelevant features. Our focus is on examining feature engineering from a data processing perspective and exploring different aspects of transforming the data into a more refined representation. Figure 8 shows a model that summarizes most of the research ideas in the field of cyber security.

A. FEATURE SELECTION ALGORITHMS

As discussed earlier, feature selection offers several advantages, including cost reduction in data acquisition and classification model training, decreased model size, enhanced classification performance, and potentially improved interpretability of classification models. Consequently, numerous algorithms have been developed to facilitate feature selection. Many studies classify feature selection methods into three categories: filter-based techniques, wrapper techniques, and embed techniques [50], [51], [203]. However, hybrid methods also exist outside of these categories.

1) FEATURE EXTRACTION

Feature extraction involves reducing the number of attributes in data by mapping high-dimensional features to a lower-dimensional feature space. This transformation retains the essential characteristics of the original features and can be seen as a combination of linear or non-linear features. The resultant feature space exhibits properties similar to the original features [188]. A feature selection method refers to the technique of choosing pertinent features from a given dataset. Both feature selection and feature extraction play a crucial role in enhancing learning models' performance and computational efficiency. Therefore, both approaches can be considered effective methods for feature engineering. Feature extraction is particularly valuable for extracting features that can improve the learning algorithm's performance. However, it should be noted that feature extraction alters the inherent meaning of the features, which can complicate subsequent analysis of these features [61]. In contrast to feature extraction, feature selection preserves the original physical meaning of the features. It achieves this by choosing a subset of the most relevant features from the original feature set [162]. Feature extraction and feature selection play a dominant role in the feature engineering process as they can enhance the efficiency and interpretability of learning models. By improving learning efficiency, reducing computational costs, and

TABLE 5. Compassion among datasets.

Dataset	Realistic Traffic	Label Data	IoT Traces	Zero-day Attacks	Full Packet	Captured Year
DARPA 98	✓	✓	X	X	✓	1998
KDDCUP 99	✓	✓	X	X	✓	1999
CAIDA	✓	X	X	X	X	2007
NSL-KDD	✓	✓	X	X	✓	2009
ISCX 2012	✓	✓	X	X	✓	2012
ADFA-WD	✓	✓	X	✓	✓	2014
ADFA-LD	✓	✓	X	✓	✓	2014
CICIDS2017	✓	✓	X	✓	✓	2017
Bot-IoT	✓	✓	✓	✓	✓	2018
Kyoto 2006+	✓	✓	X	X	✓	2006
CSE-CIC-IDS2018	✓	✓	X	✓	✓	2018
NF-BoT-IoT-v2	✓	✓	✓	✓	✓	2021
NF-ToN-IoT-v2	✓	✓	✓	✓	✓	2021
IoTDS20	✓	✓	✓	✓	✓	2020

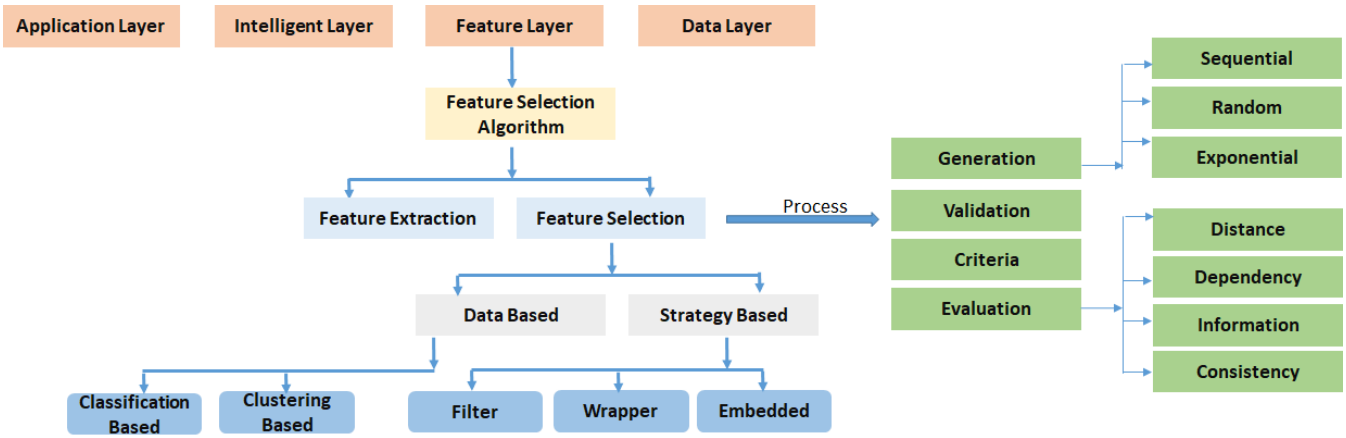

FIGURE 8. Feature selection algorithm and process based on summarized research Idea in the field of cyber security.

mitigating overfitting of data, these techniques contribute significantly to the overall effectiveness of application models.

2) FEATURE SELECTION

The feature selection method involves obtaining a subset of features from the original set of available features. Novaković [179] proposed a framework to illustrate the feature selection process, which includes selection criteria, evaluation criteria, and the learning techniques used. The suitability of the obtained features is assessed using evaluation criteria that encompass measures such as distance, information, dependence, and consistency [11], [35], [228]. The size of the problem domain increases proportionally with the number of features, and the issue of feature selection is widely considered to be NP-hard [179]. A typical feature selection process can be outlined as follows: generate an optimal subset of features, evaluate the generated subset of features, establish a termination criterion, and validate the results obtained from the selected feature set [11].

This model deals with security issues through four steps, including data selection and acquisition, data feature extraction, model construction, and specific applications. To this end, the entire model is divided into four levels as follows:

a: DATA LAYER

Data selection is a fundamental task that significantly impacts the performance of a model. In the context of the four research aspects, the data utilized in the experiments consist of both general datasets and datasets collected specifically for the research.

b: FEATURE LAYER

Accurate identification of security issues heavily relies on effective feature extraction. Prior to commencing data

extraction, it is crucial to perform unified processing of the data, particularly when working with self-collected datasets (e.g., [246]). Certain methods incorporate feature extraction within the model construction and representation process, while others conduct feature extraction as a separate step to enhance the capacity for expressing data.

c: INTELLIGENT LAYER

This layer consists of two distinct steps: modelling and evaluation. The model construction step plays a vital role in representing artificial intelligence (AI) and serves as the fundamental component for both basic methods and specific use cases. The effectiveness of the model is assessed using evaluation methods, with accuracy rate being the most commonly used measure, followed by the equal error rate (EER). Additionally, certain studies employ specialized evaluation methods tailored to address specific problems, such as response time (e.g., [146]), receiver operating characteristic (ROC) curve (e.g., [264]) etc.

d: APPLICATION LAYER

Following the model construction, these models were either utilized to address specific problems or deployed in conjunction with specific scenarios. The common thread among these applications was the utilization of AI to ensure cybersecurity. These summaries encompassed details such as datasets, features, extraction methods, classification models, and the highest achieved accuracy of the methods. Additionally, the timeliness and complexity of the methods were used for comparison. These indicators effectively gauged the efficacy of the methods and aligned with the processing requirements of cybersecurity issues.

In the field of cybersecurity, AI holds significant potential; however, it necessitates adjustments to align it more effectively with the specific demands of this domain. Current research in this field is focused on achieving rapid detection, enhancing detection accuracy, and uncovering data characteristics. These areas remain the focal points of ongoing investigations in the cybersecurity domain. Table 6 summarize some of the innovative methods, comparative analysis of classification techniques, feature selection and ML applied to various attack type.

VIII. IDS EVASION TECHNIQUES

In this discussion, we will explore various techniques that cybercriminals may employ to evade detection by intrusion detection systems (IDS), posing a challenge for such systems.

1) FRAGMENTATION

Fragmentation is an Intrusion Detection System (IDS) evasion technique that aims to bypass or deceive the detection capabilities of an IDS by splitting network traffic into smaller fragments. The technique takes advantage of how IDSs and network protocols handle fragmented packets.

When transmitted over a network, data is divided into smaller units called packets. Due to network limitations or deliberate actions, packets may be fragmented into smaller pieces to be transmitted across the network. This fragmentation process involves breaking the original packet into multiple smaller fragments, each with its header.

In the context of IDS evasion, an attacker may deliberately fragment a network packet to evade detection by an IDS. Here's how the fragmentation technique works:

1) Packet Fragmentation: The attacker crafts a network packet, typically with malicious content or payload, and fragments it into smaller pieces. The fragmentation process involves dividing the packet into smaller fragments and assigning a header to each fragment.
2) Transmission: The attacker then transmits these fragmented packets over the network. Each packet fragment is sent individually and may take different routes to reach the destination.
3) Reassembly: At the receiving end, the network stack is responsible for reassembling the fragmented packets to reconstruct the original packet. The network stack uses information from the headers of each fragment to reconstruct the packet properly.
4) IDS Detection: An IDS inspects network traffic and identifies potential malicious activity or anomalies. However, due to the fragmentation technique, the IDS may face challenges in accurately inspecting and analyzing the fragmented packets. The IDS may analyze each fragment separately, without the full context of the original packet, leading to potential evasion.
5) Evasion of IDS: By fragmenting the packet, the attacker may distribute the malicious payload across multiple fragments, making it difficult for the IDS to detect the full extent of the malicious content. The attacker may exploit vulnerabilities in the IDS's packet reassembly process, timing limitations, or weaknesses in the IDS's handling of fragmented packets to evade detection.

To counter fragmentation-based evasion techniques, IDSs need to implement robust packet reassembly algorithms that can accurately reconstruct and analyse fragmented packets. Additionally, IDSs may employ techniques such as heuristic analysis, anomaly detection, or behaviour-based detection to identify potential evasion attempts and suspicious network activities [191], [130].

2) FLOODING

Flooding is an IDS (Intrusion Detection System) evasion technique that aims to overwhelm or flood the system with high network traffic or events, making it difficult for the IDS to analyze and effectively detect potential intrusions or attacks.

In a flooding attack, an attacker intentionally generates many network packets or events, often exceeding the system's processing capacity. This flood of traffic can take various forms, such as TCP/IP packets, ICMP (Internet Control Message Protocol) messages, or application-layer requests. The attacker often uses techniques such as spoofing real User Datagram Protocol (UDP) [52] and Internet Control Message

TABLE 6. Different innovative methods, comparative analysis of classification techniques, feature selection and ML applied to various attack types.

Author	Dataset	Feature Selection	Classification Technique	Attack Detected	Detection Method	Remarks
Zhang et al. [265]	KDDCUP99	Information Gain (IG)	Random Forest	DoS, R2L, U2R, Probe	RF, Outlier Detection	Accuracy and complexityHigh
Shafiq et al. [207]	Bot-IoT	Relief-F and Pearson's Correlation Coefficient	RF,J48 and MLP	Botnet attacks	RF, Outlier Detection	High accuracy,extended as DL classifiers
Marir et al. [153]	KDDCup99, CI-CIDS2017	Multi-layer SVM	Distributed DBN	DoS, R2L, U2R, Probe	DBN, Ensemble SVM	High accuracy and Complexity
Bhati et al. [45]	NSL-KDD	NIDS	SVM	DoS, Probe, R2L, U2R	L-SVM, Q-SVM, FG-SVM	Accuracy and Complexity Mid
Bbhatia et al [46]	Public datasets	Optimal regression-based approach	SVM, gradient boosting, RF	DDoS, Botnet attacks	NFD based ML	Mid Accuracy and Complexity
Kabir et al [114]	KDDCUP99	Optimum Allocation	Anomaly, Misuse	Dos, R2L, U2R, Probe	LS-SVM	High Accuracy and Complexity
Miamo et al [150]l	CTU	5G anomaly symptom and network	DNN LSTM	Botnet	LSTM	Low accuracy for high traffic
Shorman et al [29]	NN-BaIoT	Isolation Forest	LOF, OCSVM	Botnet	GWO-OCSVM	High Accuracy
Mafarja et al [149]	D	Ensemble of trees	Random Subspace and Random Tree (RSRT)	Cyber-attack detection of SCADA	ML	Improved security, high exclusion time
Thakkar et al [228]	NSLKDD	Chi-Square, RFE, IG	SVM	DoS, Probe, R2L, U2R	DT, RF, LR, k-NN, SVM, NB, ANN	SVM with RFE outperformed other classifiers
Meftah et al [154]	UNSW-NB15	RFE	CT	DoS, exploits, worms, backdoor, shellcode	SVM, stocastic gradient decent, LR	Accuracy (DT) is Good
Zhang et al [269]	CICIDS2017	Flow features	PCCN	Web,SSH, DOS,Portscan attack etc	PCCN	Accuracy high

Protocol (ICMP) [94] to induce flooding. The attacker's objective is to consume the system's resources, exhaust its processing capabilities, or create confusion and obfuscation, thereby evading detection by the IDS.

Flooding attacks can exploit network protocol vulnerabilities or target specific IDS components. Here are a few common types of flooding attacks:

1) Denial of Service (DoS) Flooding: In this attack, the attacker floods the targeted network or system with massive traffic, overwhelming its capacity to respond to legitimate requests. By consuming the available resources, the attacker can cause disruptions or temporary unavailability of the targeted system, diverting attention from other malicious activities.
2) Distributed Denial of Service (DDoS) Flooding: DDoS attacks involve multiple compromised systems (often a botnet) coordinating the flooding attack simultaneously. By distributing the attack across many sources, blocking or mitigating the flood of traffic becomes more challenging, increasing the impact on the targeted system.
3) Protocol-Level Flooding: These attacks exploit network protocol vulnerabilities, such as TCP/IP or ICMP. By generating a large volume of malformed or specially crafted packets, the attacker aims to overwhelm the protocol handlers within the IDS, potentially causing system crashes or bypassing certain security measures.

To counteract flooding attacks, IDSs employ various defense mechanisms. These may include traffic filtering, rate limiting, traffic prioritization, or anomaly detection algorithms that can identify abnormal traffic patterns associated with flooding attacks. Network administrators and security professionals can also implement firewalls, intrusion prevention systems (IPS), or traffic analysis tools to detect and mitigate flooding attacks [103], [235].

3) OBFUSCATION

In the context of Intrusion Detection Systems (IDS), obfuscation refers to a technique employed by attackers to evade detection by obscuring or altering the characteristics of malicious activities or data. Obfuscation aims to make the malicious behaviour or payload less recognizable or understandable to the IDS, thus reducing the chances of triggering an alarm or raising suspicion.

Here are a few common methods of obfuscation used to evade IDS:

1) Encryption: Attackers may encrypt their malicious code or payloads to make them appear as harmless

or encrypted data during transmission. This makes it difficult for the IDS to inspect the content and identify malicious intent.
2) Polymorphism: Polymorphic techniques involve altering the code or payload of malware on-the-fly to generate different variants that retain the same functionality but possess different signatures. This makes it challenging for signature-based IDS systems to detect and match against known patterns.
3) Code Obfuscation: Attackers may intentionally obfuscate their code by using techniques such as code packing, variable renaming, or inserting irrelevant or misleading code snippets. This makes the code more difficult to analyze and understand, potentially bypassing signature-based detection mechanisms.
4) Protocol Tunneling: Attackers may encapsulate their malicious activities within legitimate protocols or services. By tunnelling their traffic through trusted channels, they can evade detection by IDS systems focusing on monitoring specific protocols or network traffic.
5) Traffic Fragmentation: Attackers may fragment their malicious traffic into smaller packets or spread it across multiple connections to evade pattern-based detection. This technique aims to make the malicious traffic appear more like legitimate and benign network activity, making it harder for IDS to detect malicious patterns.

Obfuscation techniques are continually evolving as attackers adapt to new defense mechanisms. To effectively counter evasion techniques, IDS systems must employ advanced detection mechanisms such as behaviour-based analysis, anomaly detection, and machine learning algorithms that can identify suspicious patterns, abnormal behaviour, or statistical deviations in network traffic or system activities. Regular updates and keeping abreast of emerging evasion techniques are also crucial to maintaining the effectiveness of IDS systems [128].

4) ENCRYPTION

Encryption is an intrusion detection system (IDS) evasion technique that aims to hide the content of network traffic from detection by encrypting it. Encryption involves transforming the original plaintext data into an unreadable form called ciphertext using an encryption algorithm and a secret key. The ciphertext can only be decrypted back to plaintext using the corresponding decryption algorithm and key.

When an attacker wants to evade detection by an IDS, they can employ encryption to protect the payload of their network traffic. By encrypting the data, the attacker ensures that the IDS cannot inspect the payload for malicious content or detect any suspicious patterns. The encrypted traffic appears as random, unreadable data to the IDS, making it difficult for the system to analyze and identify potential threats.

Using encryption as an IDS evasion technique presents challenges for intrusion detection because the IDS primarily relies on inspecting the content of network packets to identify anomalies, signatures of known attacks, or suspicious behaviour. When traffic is encrypted, the IDS cannot directly inspect the payload, as it appears as an opaque blob of encrypted data. Encrypted traffic, such as HTTPS, cannot be interpreted by an Intrusion Detection System (IDS) [159], making it difficult for the detector to compare it to the signatures stored in its database. As a result, encrypted traffic presents a challenge for the IDS to detect attacks, even though encrypted traffic can still be analyzed for certain behaviours [57].

To address this evasion technique, IDS systems often employ various methods:

1) Traffic Metadata Analysis: While the payload may be encrypted, certain metadata associated with the network traffic can still provide insights. IDS systems can analyze packet size, flow duration, source and destination addresses to identify suspicious patterns or behaviour.
2) Encrypted Traffic Analysis: Some IDS systems are designed to analyze encrypted traffic by inspecting characteristics such as packet timing, size distribution, or flow patterns. By analyzing these features, the IDS may detect anomalies or behavioural patterns that suggest malicious activity.
3) SSL/TLS Inspection: In the case of encrypted web traffic (HTTPS), IDS systems can leverage SSL/TLS inspection techniques. These methods involve decrypting the encrypted traffic at the IDS, inspecting the plaintext content for potential threats, and re-encrypting the traffic before forwarding it to the intended recipient.

It is important to note that encryption itself is a critical security measure for protecting sensitive data in transit. However, adversaries can exploit encryption as an IDS evasion technique to conceal their malicious activities. To combat such evasion techniques, IDS systems employ a combination of traffic analysis, heuristics, behavioural monitoring, and other advanced techniques to detect and mitigate potential threats, even when encryption is utilized.

5) SOURCE ROUTING

Source Routing is an IDS (Intrusion Detection System) evasion technique employed by attackers to bypass network security systems. It involves manipulating the source IP address or the routing path of network packets to avoid detection or gain unauthorized access to a target network.

In typical network communication, packets travel from a source device to a destination device following a predetermined path determined by routers and switches. However, in Source Routing-based attacks, the attacker modifies the packet headers to specify the path that the packet should follow, overriding the default routing mechanisms of the network.

There are two primary variations of Source Routing evasion techniques:

1) Loose Source Routing: In this technique, the attacker specifies a list of intermediate network nodes (routers) through which the packet should be routed. The packet includes a source route option in its header, indicating the sequence of intermediate routers. When the packet reaches each specified router, it checks the routing option and forwards the packet accordingly. By specifying a different routing path, the attacker can bypass or confuse IDS systems that rely on monitoring specific network paths.
2) Strict Source Routing: The attacker specifies the exact sequence of routers the packet must follow in this technique. The packet includes a complete source route option, providing a precise path for the packet. Each router on the path verifies the packet's source route option and forwards it accordingly. By specifying a controlled path, the attacker can potentially avoid network security measures designed to monitor or block traffic based on predefined routing paths.

By utilizing Source Routing techniques, attackers can attempt to circumvent network-based security controls, such as IDS systems that rely on monitoring network traffic or analyzing packet metadata. These techniques make it challenging for IDS systems to detect and analyze malicious activities or anomalous behaviour accurately.

To mitigate Source Routing-based evasion techniques, network administrators and security professionals can implement measures such as:

- Enforcing strict ingress and egress filtering to prevent packets with source routing options from entering or leaving the network.
- Configuring network devices to ignore or drop packets that contain source routing options.
- Monitoring and analyzing network traffic for suspicious or unauthorized source routing.
- Keeping network devices and systems updated with the latest security patches to minimize vulnerabilities that Source Routing attacks could exploit.

By implementing these preventive measures, organizations can enhance the network security posture and reduce the risk of successful intrusion attempts using Source Routing evasion techniques [224], [242].

6) SOURCE PORT MANIPULATION

Source Port Manipulation is an evasion technique commonly used in attempts to bypass Intrusion Detection Systems (IDSs) and evade detection. In network communication, each packet contains a source port and a destination port, which help identify the source and destination of the communication. Source Port Manipulation involves altering the source port value in network packets to deceive the IDS and avoid triggering any suspicious or malicious activity alerts.

The purpose of manipulating the source port is to disguise the true origin of the network traffic, making it appear as if it is originating from a different source or a legitimate service port. By changing the source port to a commonly used port number associated with a well-known service, such as port 80 for HTTP or port 443 for HTTPS, attackers attempt to blend their traffic with legitimate network traffic.

Using Source Port Manipulation, attackers hope to bypass IDSs relying on source port information to detect and block potentially malicious traffic. IDSs often monitor network traffic and apply rule-based or signature-based detection mechanisms to identify suspicious patterns or known attack signatures. However, if the source port is manipulated to mimic legitimate traffic, it becomes more challenging for the IDS to differentiate between malicious and legitimate communication.

Furthermore, Source Port Manipulation can also exploit weaknesses in IDSs that prioritize monitoring specific port numbers or services. By carefully selecting source port values that are not typically monitored or given lower priority in the IDS's rule set, attackers aim to remain undetected and slip past the IDS's detection mechanisms.

To counter Source Port Manipulation, IDSs must employ more sophisticated detection techniques considering other aspects of network traffic beyond the source port, such as the payload, traffic behaviour, or anomaly detection algorithms. Additionally, employing machine learning or anomaly detection algorithms that can identify abnormal patterns or behaviours in network traffic can help enhance the IDS's ability to detect and respond to such evasion techniques [63].

7) ADDRESS DECOY

The "Address Decoy" technique is an evasion method employed to bypass Intrusion Detection Systems (IDSs) by manipulating network addresses during a network communication session. This technique aims to deceive the IDS and evade its detection mechanisms by altering the network traffic's source or destination IP addresses.

The Address Decoy technique involves the insertion of forged or decoy IP addresses in the packet headers of network communications. By modifying the source IP address, an attacker can make it appear that the network traffic originates from a different source or location than the actual sender. Similarly, by manipulating the destination IP address, the attacker can make it seem like the traffic is intended for a different destination or target.

The purpose of employing the Address Decoy technique is to confuse the IDS and avoid triggering any alarms or alerts that may be based on specific source or destination IP addresses. Using decoy addresses, attackers can attempt to bypass IP-based filtering or detection mechanisms that rely on specific known addresses associated with malicious activities.

This evasion technique can pose a significant challenge for IDSs since it can make it more difficult for the system to attribute network traffic to its true source or destination accurately. IDSs typically rely on IP addresses to identify and

analyze network activity, so when the addresses are manipulated, it can undermine the effectiveness of the detection mechanisms.

To mitigate the Address Decoy technique, IDSs need to employ more sophisticated detection algorithms to recognize and analyze patterns beyond IP addresses. Techniques such as deep packet inspection, behaviour analysis, or anomaly detection can help identify suspicious network behaviour even when the source or destination addresses have been decoyed or forged. Additionally, employing traffic analysis techniques and considering other network attributes can enhance the detection capabilities of IDSs and reduce the effectiveness of Address Decoy evasion attempts [255].

8) RANDOMIZING THE ORDER OF HOST

The IDS evasion technique ''Randomizing the Order of Host'' is employed by attackers to evade intrusion detection by Intrusion Detection Systems (IDS). It involves manipulating the order in which network traffic or communication is initiated between hosts, intending to confuse or bypass the IDS's detection mechanisms.

The typical behaviour of network traffic involves a predictable pattern in which hosts initiate connections or communicate with each other. IDSs often rely on analyzing this pattern to detect anomalies or suspicious activities. However, by randomizing the order of hosts, attackers aim to disrupt this pattern and make their activities less conspicuous to the IDS.

By shuffling the order of hosts, an attacker can make it more challenging for the IDS to correlate network events accurately. For example, if an IDS is designed to flag multiple connection attempts from a single host as potentially malicious, randomizing the order of hosts can make it difficult for the IDS to identify and link these connection attempts together, thereby reducing the chances of detection.

Randomizing the order of hosts can be achieved through various means, such as utilizing different IP addresses, spoofing MAC addresses, or employing proxy servers or botnets to obfuscate the true source of network traffic. Attackers may also leverage techniques like IP hopping or utilizing anonymous networks to complicate the detection process further.

This evasion technique aims to exploit the limitations and assumptions of IDSs, which often rely on the predictable nature of network traffic. By introducing randomness and variability, attackers can increase their chances of evading detection and carrying out their malicious activities undetected.

IDSs need to employ more advanced and dynamic detection mechanisms to counteract this evasion technique that can adapt to changing patterns and anomalous behaviour in network traffic. This may involve incorporating machine learning algorithms, anomaly detection techniques, or behavioural analysis to identify suspicious activities regardless of the order of hosts [255].

9) SENDING THE BAD CHECKSUMS

The IDS evasion technique known as ''Sending the Bad Checksums'' is used by attackers to bypass or evade Intrusion Detection Systems (IDS). It takes advantage of how IDS systems often use checksums to verify the integrity of network packets.

A checksum is usually calculated and attached to the packet when data is transmitted over a network. The receiving system then recalculates the checksum and compares it with the one received. If the checksums match, it is assumed that the packet is intact and has not been modified during transit. However, if the checksums do not match, it indicates a potential modification or corruption of the packet.

In the ''Sending the Bad Checksums'' evasion technique, an attacker intentionally modifies the contents of a packet while leaving the checksum unchanged. By doing so, the attacker aims to trick the IDS into accepting the modified packet as legitimate and not flagging it as suspicious or malicious.

This evasion technique exploits a weakness in some IDS systems that primarily rely on checksum verification to detect anomalies or intrusions. By manipulating the packet content without affecting the checksum, the attacker attempts to evade detection by the IDS. Sending packets with incorrect or deceptive TCP/UDP checksums to the intended target can get around defense frameworks. The TCP/UDP checksums ensure the integrity of the data. As a result, sending packets with incorrect checksums makes it easier for attackers to collect data from systems with poor configuration by looking for a response [208], [255].

More sophisticated IDS systems employ additional methods beyond checksum verification to counter this evasion technique. These may include analyzing packet payloads, examining network behaviour patterns, or utilizing more advanced anomaly detection techniques. By incorporating multiple layers of detection and analysis, IDS systems can improve their resilience against evasion techniques like ''Sending the Bad Checksums.''

10) SPOOFING THE IP ADDRESS

Spoofing the IP address is an intrusion detection system (IDS) evasion technique in which an attacker manipulates or forges the source IP address of a network packet to deceive the IDS and avoid detection. The primary objective of IP address spoofing is to conceal the true identity or origin of the attacker, making it challenging for the IDS to identify and trace the source of the malicious activity accurately.

Here's how the spoofing technique works:

1) Source IP Manipulation: The attacker modifies the source IP address field in the IP header of a network packet. This can be done using various methods, such as crafting packets with custom headers or utilizing specialized software tools.
2) False Source Identification: By spoofing the source IP address, the attacker can make it appear that the network traffic originates from a different source, such

as a trusted or authorized IP address. This misleads the IDS into believing that the traffic is legitimate and originating from a trusted entity.
3) IDS Bypass: When the spoofed packets reach the IDS, they may be analyzed and processed based on the falsified source IP address. This can result in misinterpretation of the traffic, as the IDS might associate it with the falsified source rather than the attacker. As a result, the malicious activity may go undetected or be attributed to an innocent party.
4) Exploitation: Once the attacker successfully evades the IDS by spoofing the IP address, they can proceed with various malicious activities, such as launching attacks, exploiting vulnerabilities, initiating unauthorized access attempts, or conducting reconnaissance without raising suspicion from the IDS.

IP address spoofing is commonly used in several attacks, including distributed denial-of-service (DDoS) attacks, IP hijacking, session hijacking, and network scanning. By disguising the true source IP address, attackers can make it difficult for security systems to identify and respond to their malicious activities accurately.

Network administrators and security professionals implement various countermeasures to mitigate the risk of IP address spoofing, such as ingress/egress filtering, implementing authentication mechanisms, and deploying network monitoring solutions capable of detecting and flagging suspicious traffic patterns. Additionally, implementing cryptographic protocols, such as IPsec, can help ensure the integrity and authenticity of network communications by verifying the source IP address [152], [255].

11) PROXY SERVERS

Proxy servers are commonly used as an IDS (Intrusion Detection System) evasion technique. A proxy server acts as an intermediary between a client and a destination server, forwarding requests from the client to the server and relaying the responses back to the client. By utilizing proxy servers, attackers can attempt to hide their malicious activities and bypass or deceive intrusion detection systems.

When an attacker employs proxy servers as an IDS evasion technique, they typically take the following steps:

1) Concealing the Source: The attacker initiates malicious activities from their system or network and connects to a proxy server. This connection serves as a means to hide the true source of the attack. The attacker's activities appear to originate from the IP address and network of the proxy server, making it challenging for IDS systems to attribute the malicious behaviour to the actual attacker.
2) Traffic Redirection: The attacker directs their network traffic once connected to the proxy server. This can involve configuring their tools or malware to route communication through the proxy server or manually configuring network settings to redirect traffic accordingly. By doing so, the attacker's traffic passes through the proxy server, making it more difficult for IDS systems to detect and analyze malicious activities directly.
3) Encryption and Tunneling: To further obfuscate their activities, attackers may encrypt their network traffic or employ tunnelling techniques such as Virtual Private Networks (VPNs) or Secure Shell (SSH) tunnels. Encryption helps to prevent IDS systems from inspecting the content of the traffic. At the same time, tunnelling allows attackers to encapsulate their communication within a secure channel, making it harder for IDS systems to detect and interpret the traffic.

By leveraging proxy servers, attackers can attempt to evade detection by IDS systems that rely on monitoring network traffic, IP addresses, or other indicators to identify malicious activities. Using proxy servers adds additional complexity and indirection to the attacker's activities, making it challenging for IDS systems to accurately attribute and detect malicious behaviour.

To counteract this evasion technique, IDS systems can employ techniques such as traffic analysis, behaviour-based anomaly detection, or monitoring of known proxy server IP addresses. Additionally, network administrators can implement measures to block or restrict access to known malicious or anonymizing proxy servers to prevent attackers from exploiting them [250].

12) ANONYMIZERS

Anonymizers are an intrusion detection system (IDS) evasion technique that attackers employ to obfuscate their malicious activities and bypass detection mechanisms. Anonymisers aim to hide the true identity or origin of network traffic or malicious payloads, making it difficult for IDS systems to detect and analyze the attacks accurately.

Anonymizers can take different forms and leverage various methods to achieve their evasion objectives. Here are a few common techniques used by anonymizers:

1) Proxy Servers: Attackers can use proxy servers to route their network traffic through an intermediate server before reaching the target system. By doing so, the attacker's IP address is masked, and the traffic appears to originate from the proxy server. This makes it challenging for IDS systems to trace the true source of the attack [250].
2) Tor Network: The Tor (The Onion Router) network is a popular anonymization system employing relays to route network traffic through multiple nodes, obscuring the original source. The traffic is encrypted and passed through different nodes, making it challenging for IDS systems to track the attacker's location and activities.
3) VPNs (Virtual Private Networks): Attackers can utilize VPN services to establish an encrypted connection to a remote server. By tunnelling their traffic through the VPN, attackers can hide their IP address and make it appear that the traffic originates from the VPN server.
4) IP Spoofing: This technique involves forging the source IP address of network packets to make them appear

as if they come from a trusted or legitimate source. By spoofing the IP address, attackers can evade detection by IDS systems that rely on IP-based filtering or reputation-based mechanisms [152], [255].

The use of anonymizers presents significant challenges for IDS systems. They complicate the task of accurately identifying and attributing attacks, as the true origin or identity of the attacker is obfuscated. IDS systems must employ advanced techniques, such as behaviour analysis, anomaly detection, or deep packet inspection, to detect and mitigate attacks that utilize anonymizers.

To counter these evasion techniques, IDS systems may employ strategies such as monitoring for suspicious traffic patterns, analyzing packet contents beyond IP addresses, employing machine learning algorithms to detect anomalies, or collaborating with threat intelligence sources to identify known anonymizers and malicious activities associated with them [107], [160].

IX. RESEARCH CHALLENGES

A. UNAVAILABILITY OF A SYSTEMATIC DATASET

Most IDS methods proposed by researchers, nearly 80 % of them, rely on DL-based or DL-ML-based models that are highly complex and require significant processing time and computing resources. Such demands can strain the processing unit, which may negatively impact the performance of the IDS. While high-performance CPUs can reduce processing time, they can also be costly. Thus, developing an efficient feature selection algorithm is essential to intelligently identify the most important features, reduce computational complexity, and enhance processing speed.

B. LIGHTWEIGHT IDS FOR IoT

The IDS system provides security to IoT networks by collecting large amounts of crucial data through the internet using sensor nodes. However, these sensor nodes have limited computational power, storage capacity, and battery life, making installing an effective IDS system challenging. In IoT networks, IDS can be placed either at the point where network traffic from the internet enters or distributed across sensor nodes. In both cases, IDS must be effective at detecting malicious attacks. However, in the second scenario, a lightweight IDS model is required for sensor nodes with limited resources. The main challenge is to develop a lightweight IDS model that is computationally efficient, has a short training time, and has a higher detection rate.

C. CHALLENGES OF IDS FOR ICSs

Industrial Control Systems (ICSs) are composed of hardware and software components, including Supervisory Control and Data Acquisition (SCADA) systems, which process sensor data to monitor and control machinery and software tools that enable operators to manage these devices manually. The Stuxnet attack highlights the recent attacks on ICSs, considered the first known cyberwarfare weapon. The unique structure of Industrial Control Systems (ICSs) and the increased attention from attackers towards them make them a challenging target for intrusion detection systems. Unlike typical attacks, the primary objective of Stuxnet was likely aimed at ICSs. In contrast to a typical attack, the Iranian atomic program was probably Stuxnet's main target [178]. Attacks on ICSs may be state-sponsored, launched by rival companies, internal intruders with a malicious target, or even carried out by hacktivists.

D. REAL-TIME IMPLEMENTATION OF DEEP LEARNING ON MOBILE & WEARABLE DEVICES

Deep learning algorithms that are integrated into mobile and wearable technology will make data storage and transfer less complex to process. The current generation of mobile and wearable devices have memory and data acquisition limitations, which make this technique difficult to use. Deep learning requires a lot of parameter tuning and initialization, which extends computation time and makes it unsuitable for low-energy mobile devices. One approach to reducing training time and memory consumption is using mobile cloud computing platforms. This method could potentially provide techniques that could be used for real-time implementation. With this kind of implementation, the system can become self-adaptive and only need a small amount of user input when adding a new information source.

E. PRE-PROCESSING AND HYPER-PARAMETER SETTING's EVALUATION

Pre-processing and dimensional reduction are crucial steps in the process of identifying human activity. However, there is still much to learn about how pre-processing works and how it affects deep learning performance. Current studies on hyper-parameter optimization in deep learning rely on heuristics methods. However, several challenges still need to be addressed, such as optimizing the learning rate to increase computational speed and reduce model and data size, kernel reuse, filter size, computation time, memory analysis, and the learning process. To improve the efficiency of mobile-based deep learning techniques, further studies are necessary on grid search and evolutionary optimization methods. These methods can help reduce energy consumption, support dynamic and adaptive applications, and enable faster computation using mobile GPUs. These directions are crucial for future research, as stated by Ordóñez and Roggen [181].

F. SUBSTANTIAL SENSOR DATASETS TO CHECK THE EFFECTIVENESS OF DL

Large datasets can be obtained through various sensor-based Internet of Things (IoT) devices and technologies for deep learning technique training and evaluation. Recent research has focused on using deep learning for human activity recognition on mobile and wearable devices. However, benchmark datasets from traditional machine learning algorithms like OPPORTUNITY, Skoda, and WSDM have been used to

evaluate the performance of these models. To improve the performance of these models, data-gathering techniques such as cyber-physical systems and mobile crowdsourcing can be used to gather data from various sources like smart homes and mobile devices. This data can be used for applications like transportation, elderly care, monitoring, context-aware location recognition, and other critical applications. Therefore, combining these technologies to gather large datasets is essential for improving the performance of these models.

G. PUTTING DL ALGORITHMS FOR WEARABLE AND MOBILE DEVICES

Recognizing transfer learning-based activities is difficult to complete. Transfer learning uses the knowledge gained in various fields to enhance the system's performance in new ones it has not yet encountered. Reduced training time, robust and adaptable activity details, reuse of prior knowledge in new domains, and a crucial activity recognition problem are the main reasons for applying transfer learning. The implementation of deep learning-based human activity recognition will be improved by additional research in the transferability of the kernel, convolutional layer, interlocation, and intermodalities [181].

H. DL-BASED DECISION FUSION FOR RECOGNIZING HUMAN MOVEMENT IN MOBILE DEVICES

Combining multiple architectures, sensors, and classifiers into a single decision is crucial in improving the performance and diversity of human activity recognition systems. Heterogeneous sensor fusion, fusing deep learning with expert knowledge, and combining various unsupervised feature learning techniques to enhance activity recognition system performance are typical areas that need more research.

I. DL-BASED IMBALANCE ISSUE ON MOBILE DEVICES TO TRACK HUMAN ACTIVITY

Class imbalance problems are common in datasets for human activity recognition and detecting abnormal activities, particularly in healthcare monitoring for fall detection, where accurately identifying falls can be challenging. In the case of mobile and wearable-based human activity recognition, the class imbalance can be caused by distortions in the dataset or inaccuracies in sensor data calibration, reducing the generalizability of the model's performance [74]. Previous studies have proposed solutions such as mixed kernel-based weighted extreme learning machines and cost-sensitive learning strategies [252]. However, there is limited research on the impact of class imbalance on deep learning implementation for mobile and wearable sensors. Addressing class imbalance would greatly improve human activity recognition using deep learning methods.

J. AUGMENTATION TO ENHANCE DL PERFORMANCE

One open research challenge in motion sensor-based human activity recognition using convolution neural networks (CNNs) involves improving performance through data augmentation techniques. These techniques are important for generating enough training data to avoid overfitting and improving the translation in-variance to handle sensor orientation, distortion, and changes in CNN models. Data augmentation is a common training method in image classification [90]. A variety of data augmentation techniques, including shifting the locations of sensors, arbitrary rotations, permuting the locations where sensor events occurred, time warping, and scaling, will significantly improve the performance of deep learning-based human activity recognition [233].

In this section, We have discussed various Decision trees based on IDS through model evaluation.

K. RECURSIVE FEATURE ELIMINATION METHOD

Recursive Feature Elimination (RFE) is a kind of feature selection technique forming a prototype using the remaining parameters and determining its accuracy. The recursive elimination method uses the feature arrangement to forecast the desired result [71]. A feature selection technique called recursive feature elimination (RFE) eliminates a model's weakest feature (or features) until the required number of features is reached [92]. Features are ranked by the model's coefficients and by recursively eliminating a small number of features per loop. Although RFE stipulates a minimum number of features to keep, it is frequently unknown how many features are valid. Cross-validation is combined with RFE to score various feature subsets and choose the best number of features. The set of features with the highest score. The RFECV visualizer plots the number of features along with their cross-validated test score and features [174].

1) RECURSIVE FEATURE ELIMINATION (RFE) FOR DT CLASSIFIER

Cross-validation for all features using a decision tree classifier The parameters for decision tree classifiers that may produce the best results are chosen using the RFE method. This method's main objective is to sequentially remove the parameters with the lowest ranks. Here, the selected features are represented by the x-axis, and the y-axis represents the cross-validation value for those features.

By dividing the source dataset into sub-nodes based on the value test, a decision tree can learn using the divide and conquer strategy [44], [270]. And a technique known as recursive partitioning is used to repeat this process on each subset. This recursion process is completed and terminated when splitting no longer adds value to the predictions. For Decision tree data came as records [186], [219], written as:

$$(X, Y) = (X_1, X_2, X_y, L, X_k, Y) \ldots\ldots \quad (1)$$

The independent variables, represented by a vector x, consist of input variables such as X1, X2, X3, L, Xn, etc. The dependent variable, represented by Y, is treated as the target variable. The DT describes a decision-making structure represented by an ordered pair of nodes [133], [139].

Different types of Decision Trees (DTs) [141], [266], such as Classification and Regression Trees (CRT), ChiSquare Automatic Interaction Detection (CHAID), and the extended version C4.5, have sub-nodes that each contains one or more decision functions based on IF…THEN rules. The development of DT as a binary classifier aid in learning to classify new instances using a set of instances. With a specified number of iterations, the learning rate for each variant is fine-tuned to fall between 0.1 and 1 [102].

The reason for using DT based on C4.5 is that many features of the C4.5 Algorithm include its ability to handle both continuous and discrete attributes, missing values, and the production of more accurate models.

L. DT ENSEMBLE TECHNIQUE FOR IDS

The study develops ensemble learning models using AdaBoost, Bagging, Random Forest, and majority voting based on Decision Tree, using the prepared datasets [271]. The study's results are presented in Table 7, where the accuracy measure of the base classifiers used in the ensemble methods is the Decision Tree (DT).

Mounika and Rao [166] proposed a model that involves creating a system for detecting and categorizing intrusions called an Intrusion Detection and Classification System (IDCS). The system uses a single classifier called GBDT-IDS, a unified gradient-boosted decision tree, preprocessing, and data balancing techniques. The first step in the process is to perform preprocessing to remove missing symbols, unknown characters, and special letters. The next step is to balance the dataset using a technique called SMOTE, which increases the number of samples for underrepresented classes to make them equal to the number of samples for the other classes in the dataset.

M. DT BASED ON OVR (ONE-VS-REST) APPROACH

To assess our method, we look at four machine learning algorithms. We extend the binary forms of decision trees and logistic regression to the multiclass forms used in classification using OVR (One-vs-Rest). The accuracy is roughly the same for decision trees. Decision trees with OVR as shown in Table 7.

To decrease the rate of false alarms and increase accuracy, we should perform all binary classifications using a different method in the third step. We use the apiriori, FP-growth, and decision tree algorithms to achieve this. We generate 48 rules using apiriori, all of which are useless.

N. TWO-STEP BASED SVM FOLLOWED BY DT

In order to detect anomalies, a very innovative decision tree and support vector machine (SVM) approach was suggested [110]. The dataset was initially processed through a decision tree to detect anomalous content and achieve the desired outcome. The decision tree output was then fed into a support vector machine. The performance of the current system dataset was satisfactory, but the results were even more precise when compared to the improved SVM model. These experiments employed the C4.5 decision tree model, with a fixed number of leaf nodes of 2 and the pruning confidence level set at 0.25.

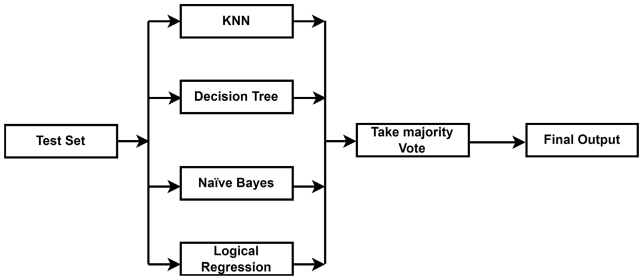

FIGURE 9. Voting classifier by combining the base classifiers used in this experiment.

To divide the forest space into distinct subsets of trees with leaf nodes corresponding to different subdivisions, a binary classifier and regression tree model was developed [113]. This was done by applying a splitting rule to each internal node. The step function performed poorly with the available dataset and generated errors.

They combined ID3, CART, and C4.5 decision trees to forecast a car's safety when fully occupied with passengers and luggage. The CART tree showed improved prediction accuracy, taking 0.5 seconds less and having a 97.36 % success rate compared to ID3 and C4.5 trees. However, it also needed more training time, resulting in over-fitting issues due to missing data. To predict student performance, a combination of DT, KNN, and SVM models was introduced [251]. SVM produced a classification report with the highest accuracy rate, which is 95 %, followed by DT with a rate of 93 % and KNN with a rate of 92 %.

O. ENSEMBLE LEARNING OR VOTING CLASSIFIER

Multiple models and classifiers are combined through the process of ensemble learning to address a specific issue. Ensemble learning uses all classifier predictions in this experiment to create a final prediction considering majority voting. In this work, k-nn, decision trees, naive bayes, and logistic regression were combined to create an ensemble model or voting classifier. The Figure 9 visualized the ensemble learning approach. When using predicted class labels for majority rule voting, this voting classifier is also known as ''hard voting'' [189].

P. C4.5 DT AND MULTILAYER PERCEPTRON (MLP) BASED HYBRID MODEL

IDS design incorporates the use of decision trees in addition to NB and unsupervised learning. Researchers have developed a decision tree and Snort-based intrusion detection method for high-speed networks [36]. A hybrid model comprising a C4.5 decision tree and Multilayer Perceptron (MCP) was trained and tested on three features of the ISCXIDS2012 dataset, achieving a detection accuracy of 99 % [93], which

TABLE 7. Decision tree comparison result.

Method	Accuracy	Precision	Recall	F-Measure
Decision tree	0.75322	0.73024	0.75322	0.71585
Decision tree OVR	0.74759	0.80435	0.74759	0.764713
Decision Tree using CHAID	0.91	X	X	X
REF for Decision tree	0.99	0.9525	0.9625	0.9575
SVM followed by DT	0.9736	X	X	X
Ensemble of 4 base classifiers	0.967	0.987	0.943	0.964

shows a detection accuracy of 99.50% and a false alarm rate as low as 0.03%. This performance is related to the author's preprocessing feature selection strategy, which was discernibility function-based. The researchers decided to create intrusion detection systems based on parallel machine learning due to the presence of fast big data networks. In a parallel computing setting, an IDS is provided by XGBoost, a state-of-the-art machine learning-based method created specifically for big data [217]. The aforementioned model has a detection rate of 99.60% and an accuracy rate of 99.65% while maintaining a low false alarm rate of 0.302%.

Q. IoT-BASED DECISION TREE

The Adaptive Multimode Decision Tree Classification Model is a novel network model that enables efficient and rapid data collection while also providing decentralized processing [116]. This model utilizes system analysis to address security concerns in 5G and IoT environments, particularly in Intrusion Detection Systems (IDS). The current data-gathering system relies on this model's adaptive and multimodal decision tree classification capabilities.

Buschlinger et al. [56] proposed method uses machine learning (ML) to develop rules for an effective rule-based Intrusion Detection System (IDS), such as Snort, which simplifies the challenging and time-consuming task of generating a rule set. Decision trees are utilized in our approach to generating rules that serve as a foundation for a rule set tailored to a particular safety-critical use case, which experts can further modify. We use long short-term memory techniques to address the issue of insufficient training data in safety-critical Automotive systems.

Le et al. [136] propose to improve the attack detection capability of IDS by using large IoT-based IDS datasets while also providing interpretability of the ML model predictions. The proposed ML-based IDS method employs an ensemble trees approach that utilizes decision tree (DT) and random forest (RF) classifiers, which do not require significant computational resources for model training. Two large datasets, NF-BoT-IoT-v2 and NF-ToN-IoT-v2 [205], along with the IoTDS20 dataset, are employed in the experimental evaluation of the proposed method, with the feature set of the net flow meter being utilized. Additionally, the SHapley additive exPlanations (SHAP) method is applied to the eXplainable AI (XAI) methodology to interpret and explain the classification decisions of the DT and RF models.

Another research presents a new approach to building a lightweight Intrusion Detection System using a unique data pre-processing technique, machine learning, and deep learning classifiers [125]. It explores different classifiers that can be used to create efficient intrusion detection systems capable of detecting Distributed Denial of Service attacks in IoT networks. The research uses two datasets, BOT-IoT, and TON-IoT Network dataset, provided by the University of New South Wales Sydney (UNSW) Australia. The datasets contain examples of DDoS attacks used for experimentation and analysis.

X. OPEN RESEARCH ISSUES

Cybercriminals have recently exhibited their adeptness in masking their identities, concealing their communication, dissociating them from illicit gains, and leveraging resilient infrastructures. Consequently, it is becoming more critical to safeguard computer systems by utilizing sophisticated intrusion detection systems that can identify modern malware. So below, we have discussed many open research sections to focus on.

A. DEFICIENCY OF DATASETS

Generating new datasets is necessary since there are limited datasets available, and some problems exist in those datasets. However, creating new datasets requires expert knowledge and is labor-intensive. Additionally, the presence of erotic content on the internet complicates the dataset shortage problem.

Moreover, machine learning techniques have the potential to detect intrusions effectively, but they may not perform well on new, unseen data. Most existing machine learning models have been trained on labelled datasets. Hence, the accuracy of the models on real-world data is not guaranteed, especially when the dataset does not include all typical real-world samples, despite achieving high accuracy on test sets.

The study analyzes the limitations and evaluation outcomes of commonly used public datasets for IDS research and discusses their data collection methods. Due to the frequent changes in normal activities and the need to cover a wide range of malware activities, there is a demand for newer and more extensive datasets.

The current machine learning techniques heavily rely on outdated datasets because publicly available datasets are the only acceptable options. While these datasets are widely used as benchmarks, they no longer reflect contemporary zero-day attacks. Although some new datasets contain many new attacks, they are still inadequate.

Therefore, testing IDS using these datasets does not provide an accurate evaluation and may lead to incorrect claims about their effectiveness. A more comprehensive data set needs to be obtained using network attacks such as Mac flooding, DHCP snooping, arp spoofing, and other modern attacks that are not used in most data sets.

B. EFFECTIVE AIDS, SIDS, AND NIDS SELECTION

Detecting hidden attacks is the main obstacle for both SIDS and AIDS. The effectiveness of IDS in detecting such attacks depends on its ability to recover the initial attack signature or generate new signatures to account for modifications made to the attack. However, IDS still needs further development to improve its robustness against various evasion techniques.

An example of the limitations of SIDS is that it can identify changes from basic mutations using regular expressions, but it may not be successful in detecting more sophisticated obfuscation methods, like encryption and packing, which attackers utilize to conceal malicious software. Similarly, NIDS is an essential defense mechanism against network intrusions, but its ability to detect zero-day attacks with low false alarms is limited.

To improve the efficiency of IDS, it is necessary to have an updated, organized, and unbiased dataset. Additionally, researchers should develop an effective NIDS framework capable of providing comprehensive security against intrusions in modern networks like IoT. The IDS framework should update the attack definitions in the dataset regularly and continuously train the model with the latest definitions to improve the IDS model's ability to detect zero-day attacks and reduce false positives.

Although the training phase of an AI-based IDS model can be time-consuming and performed offline, regular dataset updating and training are critical for accurate attack detection and IDS model efficiency.

C. CHALLENGES IN IoT ENVIRONMENT

Currently, an intelligent classifier is implemented and evaluated using various performance metrics, including packet delivery ratio, delay, average energy consumption, network lifetime, DoS attack, probe attack, L2R attack, R2L attack, and security analysis. Specifically, it is crucial to develop an intelligent IDS that can adapt to the dynamic network topology of an IoT-based network to overcome the current limitations of the detection capability. These IDSs are typically classified into categories based on anomaly detection, signature, specifications, and hybrid methods used for conducting a comparative analysis. We must assess each IDS category's performance based on several performance metrics, such as network resource optimization, false positive intrusion detection rate, and scalability.

D. SYSTEM ENVIRONMENT AND COMPLEX MODELS SOLUTION

The widespread use of IoT in various fields, such as home automation, industrial automation, and smart city systems, has led to the development of numerous microcontroller-based devices, communication protocols, and platforms. DL-based IDSs have gained popularity due to their ability to learn deep features and detect malicious attacks accurately. However, these complex models require significant computational resources, storage capacity, and time. To address this, high-performance GPUs or cloud-based GPU platforms can be used for the efficient processing of big datasets. Additionally, intelligent feature engineering can reduce the complexity of the model, resulting in almost the same detection accuracy with fewer computing resources in real-time environments.

E. USE OF STATISTICS AND DIFFERENT DL ALGORITHMS

In the field of IoT-based big data security, various machine learning methods are being used for intrusion detection. However, it is important to note that the performance of these algorithms can vary depending on the dataset and algorithm characteristics. DL-based algorithms have shown promising results for IDS design, and many have been effectively used. However, some DL algorithms, such as deep reinforcement learning and Hidden Markov Models, still require further attention. Moreover, it could be beneficial for researchers to investigate using deep learning for feature extraction in conjunction with machine learning for classification to simplify the proposed model. In general, the study of using deep learning for intrusion detection in IoT is still in its nascent stages, and more examination is required in this domain. Besides, DL is usually complex and resource-consuming, sometimes resulting in computing delays. Some DL is typically tested in the lab environment and then offered live.

XI. FUTURE WORK

As the field of Internet of Things (IoT) continues to expand, ensuring the security of IoT data becomes increasingly crucial. Numerous research papers have been published on Intrusion Detection Systems (IDSs) for securing IoT data. However, despite the progress made, several unresolved research challenges and issues persist, particularly in the realm of using Machine Learning (ML) methods for anomaly and intrusion detection in IoT for big data security. These challenges arise due to incomplete or insufficient datasets and the difficulty in satisfying all stakeholders' requirements. To advance the field further, future research should focus on addressing these challenges and exploring innovative approaches. Here are some potential research prospects that can pave the way for enhanced IoT data security:

1) Develop advanced ML methods tailored for anomaly and intrusion detection in IoT.

2) Address the challenge of incomplete or insufficient datasets through techniques such as data augmentation or synthetic data generation.
3) Explore approaches that can effectively satisfy the diverse requirements of stakeholders involved in IoT data security.
4) Investigate and evaluate the efficacy of different feature selection algorithms for improving anomaly and intrusion detection performance in IoT environments.
5) Assess the applicability and benefits of graph neural network techniques in implementing IDSs for IoT security.
6) Conduct comparative studies to compare the performance of decision tree-based approaches with neural network-based approaches in classifying filtered IoT data.
7) Develop robust classification methods capable of handling diverse IoT data types and achieving high accuracy in detecting anomalies and intrusions.
8) Design adaptive feature filtering capabilities that can dynamically adapt to evolving IoT data patterns, ensuring the relevance and effectiveness of selected features.

By addressing these research prospects, we can overcome the existing challenges and pave the way for more secure and reliable IoT data systems, benefiting a wide range of industries and applications.

XII. CONCLUSION

This paper offers a comprehensive overview of network intrusion detection mechanisms that utilize the combination of ML, DL, and DT. Its purpose is to educate new researchers on the latest advancements and trends in the field of AI-based NIDS. A systematic approach was taken in selecting relevant articles. The concept of IDS and its various classification methods are thoroughly explained based on the reviewed literature. The methodology and pros and cons of each approach are discussed in terms of their effectiveness in detecting intrusions and the complexity of the models. Improved performance and efficacy in NIDS, with higher detection accuracy and lower false alarm rates. DL-based approaches tend to outperform ML-based methods because they automatically learn features and have stronger modelling capabilities. These techniques can be challenging to implement in real-time NIDS and improve the overall performance of NIDS due to their high complexity and requirement for significant computational resources such as processing power and storage. The open research issues that require attention are discussed to provide insight into the direction of future research. Improving high computational resources will also aid in the real-time and online implementation of ML/DL on mobile and wearable devices. These ML techniques are expected to advance human activity recognition research.

REFERENCES

[1] (2023). *Dns Spoofing Attack on Brazilian Banks*. [Online]. Available: https://softwarelab.org/blog/spoofing-examples/

[2] (2015). *Google and Facebook Duped in Huge 'Scam'*. [Online]. Available: https://www.bbc.com/news/technology-39744007

[3] (2019). *Technical University Israel'S Attack*. [Online]. Available: https://www.bleepingcomputer.com/news/security/desjardins-group-data-leak-exposes-info-of-29-million-members/

[4] (2019). *Toyota Parts Supplier Hit by $ 37 Million Email Scam*. [Online]. Available: https://www.forbes.com/sites/leemathews/2019/09/06/to%20yota-parts-supplier-hit-by-37-million-email-scam/?%20sh=733a2c6e5856

[5] (2020). *Twitter Hack: Staff Tricked by Phone Spear-Phishing Scam*. [Online]. Available: https://www.bbc.com/news/technology-53607374

[6] (2021). *Debt-in Consultants Cyberattack*. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/recent-cyber-attacks

[7] (2021). *Marriott Discloses Data Breach Possibly Affecting Over 5 Million Custome*. [Online]. Available: https://www.csis.org/programs/strategic-technologies-program/significant-cyber-incidents

[8] (2021). *Saudi Aramco Confirms Data Leak After Reported Cyber Ransom*. [Online]. Available: https://www.bloomberg.com/news/articles/2021%20-07-21/saudi-aramco-confirms-data-leak-after-reported-cyber-extortion

[9] (2021). *Widespread Credential Phishing Campaign Abuses Open Redirector Link*. [Online]. Available: https://www.microsoft.com/security/bl%20og/2021/08/26/widespread-credential-phishing-campaign-abuses-open-redirector-links/

[10] (2023). *Technical University Israel's Attack*. [Online]. Available: https://www.databreaches.net/technion-university-hacked-and-locked-previously-unknown-attackers-demand-80-btc/

[11] A. Abbas, M. A. Khan, S. Latif, M. Ajaz, A. A. Shah, and J. Ahmad, "A new ensemble-based intrusion detection system for Internet of Things," *Arabian J. Sci. Eng.*, vol. 47, pp. 1–15, Aug. 2021.

[12] M. S. Abdalzaher and O. Muta, "Employing game theory and TDMA protocol to enhance security and manage power consumption in WSNs-based cognitive radio," *IEEE Access*, vol. 7, pp. 132923–132936, 2019.

[13] M. S. Abdalzaher and O. Muta, "A game-theoretic approach for enhancing security and data trustworthiness in IoT applications," *IEEE Internet Things J.*, vol. 7, no. 11, pp. 11250–11261, Nov. 2020.

[14] M. S. Abdalzaher, K. Seddik, and O. Muta, "An effective Stackelberg game for high-assurance of data trustworthiness in WSNs," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jul. 2017, pp. 1257–1262.

[15] M. S. Abdalzaher, K. Seddik, and O. Muta, "Using repeated game for maximizing high priority data trustworthiness in wireless sensor networks," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jul. 2017, pp. 552–557.

[16] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Inf. Fusion*, vol. 76, pp. 243–297, Dec. 2021.

[17] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdiscipl. Rev., Comput. Statist.*, vol. 2, no. 4, pp. 433–459, 2010.

[18] A. A. Aburomman and M. B. Ibne Reaz, "A novel SVM-kNN-PSO ensemble method for intrusion detection system," *Appl. Soft Comput.*, vol. 38, pp. 360–372, Jan. 2016.

[19] Australian Cyber Security Centre (ACSC). (2017). *ACSC Threat Report 2017*. [Online]. Available: https://www.cyber.gov.au/sites/default/files/2023-03/ACSC_Threat_Report_2017.pdf

[20] A. Aggarwal, M. Mittal, and G. Battineni, "Generative adversarial network: An overview of theory and applications," *Int. J. Inf. Manage. Data Insights*, vol. 1, no. 1, Apr. 2021, Art. no. 100004.

[21] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Procedia Comput. Sci.*, vol. 60, pp. 708–713, 2015.

[22] Y. Mehmood, F. Ahmad, I. Yaqoob, A. Adnane, M. Imran, and S. Guizani, "Internet-of-Things-based smart cities: Recent advances and challenges," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 16–24, 2017.

[23] F. Ahmad, Z. Ahmad, C. A. Kerrache, F. Kurugollu, A. Adnane, and E. Barka, "Blockchain in Internet-of-Things: Architecture, applications and research directions," in *Proc. Int. Conf. Comput. Inf. Sci. (ICCIS)*, Apr. 2019, pp. 1–6.

[24] F. Ahmad, F. Kurugollu, A. Adnane, R. Hussain, and F. Hussain, "MARINE: Man-in-the-middle attack resistant trust model in connected vehicles," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3310–3322, Apr. 2020.

[25] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 1, p. e4150, Jan. 2021.
[26] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *J. Netw. Comput. Appl.*, vol. 60, pp. 19–31, Jan. 2016.
[27] A. Ahmim, M. Derdour, and M. A. Ferrag, "An intrusion detection system based on combining probability predictions of a tree of classifiers," *Int. J. Commun. Syst.*, vol. 31, no. 9, p. e3547, Jun. 2018.
[28] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, "Deep learning approach combining sparse autoencoder with SVM for network intrusion detection," *IEEE Access*, vol. 6, pp. 52843–52856, 2018.
[29] A. Al Shorman, H. Faris, and I. Aljarah, "Unsupervised intelligent system based on one class support vector machine and Grey Wolf optimization for IoT botnet detection," *J. Ambient Intell. Humanized Comput.*, vol. 11, no. 7, pp. 2809–2825, Jul. 2020.
[30] S. Alam and N. Yao, "The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis," *Comput. Math. Org. Theory*, vol. 25, no. 3, pp. 319–335, Sep. 2019.
[31] C. Alcaraz, "Cloud-assisted dynamic resilience for cyber-physical control systems," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 76–82, Feb. 2018.
[32] K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," in *Proc. 15th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2016, pp. 195–200.
[33] B. Alsughayyir, A. M. Qamar, and R. Khan, "Developing a network attack detection system using deep learning," in *Proc. Int. Conf. Comput. Inf. Sci. (ICCIS)*, Apr. 2019, pp. 1–5.
[34] M. Alzantot, S. Chakraborty, and M. Srivastava, "SenseGen: A deep learning architecture for synthetic sensor data generation," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2017, pp. 188–193.
[35] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 2986–2998, Oct. 2016.
[36] A. Ammar, "A decision tree classifier for intrusion detection priority tagging," *J. Comput. Commun.*, vol. 3, no. 4, pp. 52–58, 2015.
[37] G. Andresini, A. Appice, N. D. Mauro, C. Loglisci, and D. Malerba, "Multi-channel deep feature learning for intrusion detection," *IEEE Access*, vol. 8, pp. 53346–53359, 2020.
[38] R. A. R. Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for intrusion detection system," *Inf. Sci.*, vol. 378, pp. 484–497, Feb. 2017.
[39] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Citeseer, 2000.
[40] K. Bajaj and A. Arora, "Dimension reduction in intrusion detection features using discriminative machine learning approach," *Int. J. omput. Sci. Issues*, vol. 10, no. 4, p. 324, 2013.
[41] K. Baker, "Singular value decomposition tutorial," The Ohio State Univ., Columbus, OH, USA, vol. 24, Tech. Rep., 2005.
[42] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "GAN dissection: Visualizing and understanding generative adversarial networks," 2018, *arXiv:1811.10597*.
[43] A. Ben-Israel and C. Iyigun, "Probabilistic D-clustering," *J. Classification*, vol. 25, pp. 5–26, Jun. 2008.
[44] N. Bhargava, G. Sharma, R. Bhargava, and M. Mathuria, "Decision tree analysis on J48 algorithm for data mining," *Proc. Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 6, pp. 1–5, 2013.
[45] B. S. Bhati and C. S. Rai, "Analysis of support vector machine-based intrusion detection techniques," *Arabian J. Sci. Eng.*, vol. 45, no. 4, pp. 2371–2383, Apr. 2020.
[46] R. Bhatia, S. Benno, J. Esteban, T. V. Lakshman, and J. Grogan, "Unsupervised machine learning for network-centric anomaly detection in IoT," in *Proc. 3rd ACM CoNEXT Workshop Big DAta, Mach. Learn. Artif. Intell. Data Commun. Netw.*, Dec. 2019, pp. 42–48.
[47] S. Bhattacharya and N. D. Lane, "From smart to deep: Robust activity recognition on smartwatches using deep learning," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2016, pp. 1–6.
[48] M. Blatt, S. Wiseman, and E. Domany, "Data clustering using a model granular magnet," *Neural Comput.*, vol. 9, no. 8, pp. 1805–1842, Nov. 1997.
[49] J. J. Blount, D. R. Tauritz, and S. A. Mulder, "Adaptive rule-based malware detection employing learning classifier systems: A proof of concept," in *Proc. IEEE 35th Annu. Comput. Softw. Appl. Conf. Workshops*, Jul. 2011, pp. 110–115.
[50] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "A review of feature selection methods on synthetic data," *Knowl. Inf. Syst.*, vol. 34, no. 3, pp. 483–519, Mar. 2013.
[51] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, J. M. Benítez, and F. Herrera, "A review of microarray datasets and applied feature selection methods," *Inf. Sci.*, vol. 282, pp. 111–135, Oct. 2014.
[52] D. Boro, H. Basumatary, T. Goswami, and D. K. Bhattacharyya, "UDP flooding attack detection using information metric measure," in *Proc. Int. Conf. ICT Sustain. Develop. (ICTSD)*. Cham, Switzerland: Springer, vol. 1, pp. 143–153, 2016.
[53] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, Aug. 1996.
[54] R. Bro and A. K. Smilde, "Principal component analysis," *Anal. Methods*, vol. 6, no. 9, pp. 2812–2831, 2014.
[55] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.
[56] L. Buschlinger, R. Rieke, S. Sarda, and C. Krauß, "Decision tree-based rule derivation for intrusion detection in safety-critical automotive systems," in *Proc. 30th Euromicro Int. Conf. Parallel, Distrib. Network-based Process. (PDP)*, Mar. 2022, pp. 246–254.
[57] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 266–282, 1st Quart., 2014.
[58] I. Cadez and P. Smyth, "Probabilistic clustering using hierarchical models," 1999.
[59] O. Can and O. K. Sahingoz, "A survey of intrusion detection systems in wireless sensor networks," in *Proc. 6th Int. Conf. Model., Simul., Appl. Optim. (ICMSAO)*, May 2015, pp. 1–6.
[60] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.
[61] U. Carrasquilla, "Benchmarking algorithms for detecting anomalies in large datasets," *MeasureIT*, pp. 1–16, Nov. 2010.
[62] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," *Comput. Secur.*, vol. 24, no. 4, pp. 295–307, Jun. 2005.
[63] P. Chemburkar. (2023). *Bypassing Firewalls: Going Beyond Source Port Manipulation*. [Online]. Available: https://payatu.com/blog/source-port-manipulation
[64] H. Chen, H. Wu, J. Hu, and C. Gao, "Event-based trust framework model in wireless sensor networks," in *Proc. Int. Conf. Netw., Archit., Storage*, Jun. 2008, pp. 359–364.
[65] J. Chung, "On deep multiscale recurrent neural networks," Ph.D. thesis, Dept. Comput. Sci. Oper. Res., Université de Montreal, Montreal, QC, Canada, 2019.
[66] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.
[67] G. Creech, "Developing a high-accuracy cross platform host-based intrusion detection system capable of reliably detecting zero-day attacks," Ph.D. thesis, Dept. Eng. Inf. Technol., UNSW Sydney, Sydney, NSW, Australia, 2014.
[68] G. Creech and J. Hu, "A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns," *IEEE Trans. Comput.*, vol. 63, no. 4, pp. 807–819, Apr. 2014.
[69] K. A. Da Costa, J. P. Papa, C. O. Lisboa, and V. H. C. de Albuquerque, "Internet of Things: A survey on machine learning-based intrusion detection approaches," *Comput. Netw.*, vol. 151, pp. 147–157, Jan. 2019.
[70] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 8609–8613.
[71] E. D. Dorothy, "An intrusion-detection model," *IEEE Trans. Softw. Eng.*, vol. 13, no. 2, pp. 222–232, Feb. 1987.
[72] X. Ding, H. Lei, and Y. Rao, "Sparse codes fusion for context enhancement of night video surveillance," *Multimedia Tools Appl.*, vol. 75, no. 18, pp. 11221–11239, Sep. 2016.

[73] S. Dua and X. Du, *Data Mining and Machine Learning in Cybersecurity*. Boca Raton, FL, USA: CRC Press, 2016.

[74] M. Edel and E. Köppe, "Binarized-BLSTM-RNN based human activity recognition," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Oct. 2016, pp. 1–7.

[75] S. Elhag, A. Fernández, A. Bawakid, S. Alshomrani, and F. Herrera, "On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on intrusion detection systems," *Exp. Syst. Appl.*, vol. 42, no. 1, pp. 193–202, Jan. 2015.

[76] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in *Proc. 20th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2018, p. 1.

[77] D. Md. Farid, N. Harbi, and M. Zahidur Rahman, "Combining naive Bayes and decision tree for adaptive intrusion detection," 2010, *arXiv:1005.4496*.

[78] A. Fischer and C. Igel, "Training restricted Boltzmann machines: An introduction," *Pattern Recognit.*, vol. 47, pp. 25–39, Jan. 2014.

[79] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "MAWILab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking," in *Proc. 6th Int. Conf.*, Nov. 2010, pp. 1–12.

[80] J. Gao, J. Yang, G. Wang, and M. Li, "A novel feature extraction method for scene recognition based on centered convolutional restricted Boltzmann machines," *Neurocomputing*, vol. 214, pp. 708–717, Nov. 2016.

[81] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An adaptive ensemble machine learning model for intrusion detection," *IEEE Access*, vol. 7, pp. 82512–82521, 2019.

[82] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Comput. Secur.*, vol. 28, nos. 1–2, pp. 18–28, Feb. 2009.

[83] M. Goldstein, "FastLOF: An expectation-maximization based local outlier detection algorithm," in *Proc. 21st Int. Conf. Pattern Recognit. (ICPR)*, Nov. 2012, pp. 2282–2285.

[84] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[85] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.

[86] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[87] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," 2014, *arXiv:1412.5068*.

[88] Y. Gu, D. Li, Y. Kamiya, and S. Kamijo, "Integration of positioning and activity context information for lifelog in urban city area," *J. Inst. Navigat.*, vol. 67, no. 1, pp. 163–179, Mar. 2020.

[89] C. Guo, Y. Ping, N. Liu, and S.-S. Luo, "A two-level hybrid approach for intrusion detection," *Neurocomputing*, vol. 214, pp. 391–400, Nov. 2016.

[90] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: A review," *Neurocomputing*, vol. 187, pp. 27–48, Apr. 2016.

[91] D. Gupta, P. S. Joshi, A. K. Bhattacharjee, and R. S. Mundada, "IDS alerts classification using knowledge-based evaluation," in *Proc. 4th Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2012, pp. 1–8, doi: 10.1109/COMSNETS.2012.6151339.

[92] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Mach. Learn.*, vol. 46, pp. 389–422, Feb. 2002.

[93] A. Akyol, M. Hacibeyoglu, and B. Karlik, "Design of multilevel hybrid classifier with variant feature sets for intrusion detection system," *IEICE Trans. Inf. Syst.*, vol. 99, no. 7, pp. 1810–1821, 2016.

[94] H. Harshita, "Detection and prevention of ICMP flood DDOS attack," *Int. J. New Technol. Res.*, vol. 3, Mar. 2017, Art. no. 263333.

[95] K. Haseeb, A. Almogren, N. Islam, I. Ud Din, and Z. Jan, "An energy-efficient and secure routing protocol for intrusion avoidance in IoT-based WSN," *Energies*, vol. 12, no. 21, p. 4174, Nov. 2019.

[96] J. Healey, "The U.S. government and zero-day vulnerabilities," *J. Int. Affairs*, vol. 1, Nov. 2016. [Online]. Available: https://jia.sipa.columbia.edu/online-articles/healey_vulnerability_equities_process

[97] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," in *Neural Networks: Tricks of the Trade*. Cham, Switzerland: Springer, 2012, pp. 599–619.

[98] G. E. Hinton and T. J. Sejnowski, "Learning and relearning in Boltzmann machines," *Parallel Distrib. Process., Explor. Microstructure Cognition*, vol. 1, pp. 282–317, Feb. 1986.

[99] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[100] M. S. Hoque, M. A. Mukit, and M. A. N. Bikas, "An implementation of intrusion detection system using genetic algorithm," 2012, *arXiv:1204.1336*.

[101] J. Hortelano, J. C. Ruiz, and P. Manzoni, "Evaluating the usefulness of watchdogs for intrusion detection in VANETs," in *Proc. IEEE Int. Conf. Commun. Workshops*, May 2010, pp. 1–5.

[102] M.-L. Huang and Y.-Y. Hsu, "Fetal distress prediction using discriminant analysis, decision tree, and artificial neural network," *J. Biomed. Sci. Eng.*, vol. 5, no. 9, pp. 526–533, 2012.

[103] I. Hussain, S. Djahel, Z. Zhang, and F. Naït-Abdesselam, "A comprehensive study of flooding attack consequences and countermeasures in session initiation protocol (SIP)," *Secur. Commun. Netw.*, vol. 8, no. 18, pp. 4436–4451, Dec. 2015.

[104] E. P. Ijjina and C. K. Mohan, "Hybrid deep neural network model for human action recognition," *Appl. Soft Comput.*, vol. 46, pp. 936–952, Sep. 2016.

[105] Y. Imamverdiyev and F. Abdullayeva, "Deep learning method for denial of service attack detection based on restricted Boltzmann machine," *Big Data*, vol. 6, no. 2, pp. 159–169, Jun. 2018.

[106] M. A. Iniesta-Bonillo, R. Sánchez-Fernández, and D. Jiménez-Castillo, "Sustainability, value, and satisfaction: Model testing and cross-validation in tourist destinations," *J. Bus. Res.*, vol. 69, no. 11, pp. 5002–5007, Nov. 2016.

[107] LivingInternet. (2001). *How Anonymizers Work*. [Online]. Available: https://www.livinginternet.com/i/is_anon_work.htm

[108] M. H. Ali, B. A. D. Al Mohammed, A. Ismail, and M. F. Zolkipli, "A new intrusion detection system based on fast learning network and particle swarm optimization," *IEEE Access*, vol. 6, pp. 20255–20261, 2018.

[109] M. A. Jabbar, R. Aluvalu, and S. S. Reddy S, "RFAODE: A novel ensemble intrusion detection system," *Proc. Comput. Sci.*, vol. 115, pp. 226–234, 2017.

[110] J. Jha and L. Ragha, "Intrusion detection system using support vector machine," *Int. J. Appl. Inf. Syst.*, vol. 3, pp. 25–30, Feb. 2013.

[111] Y. Jia, M. Wang, and Y. Wang, "Network intrusion detection algorithm based on deep neural network," *IET Inf. Secur.*, vol. 13, no. 1, pp. 48–53, Jan. 2019.

[112] K. Jiang, W. Wang, A. Wang, and H. Wu, "Network intrusion detection combined hybrid sampling with deep hierarchical network," *IEEE Access*, vol. 8, pp. 32464–32476, 2020.

[113] J. F. Charles Joseph, B.-S. Lee, A. Das, and B.-C. Seet, "Cross-layer detection of sinking behavior in wireless ad hoc networks using SVM and FDA," *IEEE Trans. Dependable Secure Comput.*, vol. 8, no. 2, pp. 233–245, Mar. 2011.

[114] E. Kabir, J. Hu, H. Wang, and G. Zhuo, "A novel statistical technique for intrusion detection systems," *Future Gener. Comput. Syst.*, vol. 79, pp. 303–318, Feb. 2018.

[115] P. Kabiri and A. A. Ghorbani, "Research on intrusion detection and response: A survey," *Int. J. Netw. Secur.*, vol. 1, no. 2, pp. 84–102, 2005.

[116] P. T. Kalaivaani, R. Krishnamoorthy, A. S. Reddy, and A. D. D. Chelladurai, "Adaptive multimode decision tree classification model using effective system analysis in IDS for 5G and IoT security issues," in *Secure Communication for 5G and IoT Networks*. Springer, 2022, pp. 141–158.

[117] G. Karatas, O. Demir, and O. K. Sahingoz, "Increasing the performance of machine learning-based IDSs on an imbalanced and up-to-date dataset," *IEEE Access*, vol. 8, pp. 32150–32162, 2020.

[118] S. Keele et al., "Guidelines for performing systematic literature reviews in software engineering," Tech. Rep. EBSE 2007-001, Version 2.3, 2007.

[119] P. S. Kenkre, A. Pai, and L. Colaco, "Real time intrusion detection and prevention system," in *Proc. 3rd Int. Conf. Frontiers Intell. Comput., Theory Appl. (FICTA)*. Cham, Switzerland: Springer, 2015, pp. 405–411.

[120] J. O. Kephart, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–45, Jan. 2003.

[121] F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, "A novel two-stage deep learning model for efficient network intrusion detection," *IEEE Access*, vol. 7, pp. 30373–30385, 2019.

[122] S. Khan, A. Gani, A. W. A. Wahab, and P. K. Singh, "Feature selection of denial-of-service attacks using entropy and granular computing," *Arabian J. Sci. Eng.*, vol. 43, no. 2, pp. 499–508, Feb. 2018.

[123] S. S. Khan and B. Taati, "Detecting unseen falls from wearable devices using channel-wise ensemble of autoencoders," *Exp. Syst. Appl.*, vol. 87, pp. 280–290, Jan. 2017.

[124] Z. A. Khan and P. Herrmann, "A trust based distributed intrusion detection mechanism for Internet of Things," in *Proc. IEEE 31st Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, Mar. 2017, pp. 1169–1176.

[125] S. A. Khanday, H. Fatima, and N. Rakesh, "Implementation of intrusion detection model for DDoS attacks in lightweight IoT networks," *Expert Syst. Appl.*, vol. 215, Apr. 2023, Art. no. 119330.

[126] A. Khraisat, I. Gondal, and P. Vamplew, "An anomaly intrusion detection system using C5 decision tree classifier," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, 2018, pp. 149–155.

[127] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, pp. 1–22, Jul. 2019.

[128] D. Kim, A. Majlesi-Kupaei, and J. Roy, "DynODet: Detecting dynamic obfuscation in malware," in *Proc. Int. Conf. Detection Intrusions Malware, Vulnerability Assessment*. Cham, Switzerland: Springer, 2017, pp. 97–118.

[129] T.-T.-H. Le, J. Kim, and H. Kim, "An effective intrusion detection classifier using long short-term memory with gradient descent optimization," in *Proc. Int. Conf. Platform Technol. Service (PlatCon)*, Feb. 2017, pp. 1–6.

[130] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 184–208, 1st Quart., 2016.

[131] Krzysztoń and M. Marks, "Simulation of watchdog placement for cooperative anomaly detection in Bluetooth mesh intrusion detection system," *Simul. Model. Pract. Theory*, vol. 101, May 2020, Art. no. 102041.

[132] N. Kshetri and J. Voas, "Hacking power grids: A current problem," *Computer*, vol. 50, no. 12, pp. 91–95, Dec. 2017.

[133] T. M. Lakshmi, A. Martin, R. M. Begum, and V. P. Venkatesan, "An analysis on performance of decision tree algorithms using student's qualitative data," *Int. J. Modern Educ. Comput. Sci.*, vol. 5, no. 5, pp. 18–27, Jun. 2013.

[134] S. Lawrence, C. L. Giles, A. Chung Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 98–113, Jan. 1997.

[135] M. T. Lazarescu and L. Lavagno, "Wireless sensor networks," in *Handbook of Hardware/Software Codesign*. Cham, Switzerland: Springer, 2017, pp. 1261–1302.

[136] T.-T.-H. Le, H. Kim, H. Kang, and H. Kim, "Classification and explanation for intrusion detection system based on ensemble trees and SHAP method," *Sensors*, vol. 22, no. 3, p. 1154, Feb. 2022.

[137] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[138] Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, and K. Dai, "An efficient intrusion detection system based on support vector machines and gradually feature removal method," *Exp. Syst. Appl.*, vol. 39, no. 1, pp. 424–430, Jan. 2012.

[139] Y. Li, H.-J. Xing, Q. Hua, X.-Z. Wang, P. Batta, S. Haeri, and L. Trajkovic, "Classification of BGP anomalies using decision trees and fuzzy rough sets," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2014, pp. 1312–1317.

[140] H. J. Liao and C. H. R. Lin, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, pp. 16–24, Feb. 2013.

[141] C.-L. Lin and C.-L. Fan, "Evaluation of CART, CHAID, and QUEST algorithms: A case study of construction defects in Taiwan," *J. Asian Archit. Building Eng.*, vol. 18, no. 6, pp. 539–553, Nov. 2019.

[142] L. Lin, K. Wang, W. Zuo, M. Wang, J. Luo, and L. Zhang, "A deep structured model with radius–margin bound for 3D human activity recognition," *Int. J. Comput. Vis.*, vol. 118, no. 2, pp. 256–273, Jun. 2016.

[143] W.-C. Lin, S.-W. Ke, and C.-F. Tsai, "CANN: An intrusion detection system based on combining cluster centers and nearest neighbors," *Knowledge-Based Syst.*, vol. 78, pp. 13–21, Apr. 2015.

[144] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: A survey," *Appl. Sci.*, vol. 9, no. 20, p. 4396, Oct. 2019.

[145] X. Liu, A. Gherbi, W. Li, and M. Cheriet, "Multi features and multi-time steps LSTM based methodology for bike sharing availability prediction," *Proc. Comput. Sci.*, vol. 155, pp. 394–401, 2019.

[146] X. Lu, L. Xiao, T. Xu, Y. Zhao, Y. Tang, and W. Zhuang, "Reinforcement learning based PHY authentication for VANETs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3068–3079, Mar. 2020.

[147] T. F. Lunt, "Aucomated audit trail analysis," in *Proc. 11th Nat. Comput. Secur. Conf.*, Oct. 1988, p. 65.

[148] W. Ma, "Analysis of anomaly detection method for Internet of Things based on deep learning," *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 12, pp. 1–7, Dec. 2020.

[149] M. Mafarja, A. A. Heidari, M. Habib, H. Faris, T. Thaher, and I. Aljarah, "Augmented whale feature selection for IoT attacks: Structure, analysis and applications," *Future Gener. Comput. Syst.*, vol. 112, pp. 18–40, Nov. 2020.

[150] L. F. Maimó, Á. L. P. Gómez, F. J. G. Clemente, M. G. Pérez, and G. M. Pérez, "A self-adaptive deep learning-based system for anomaly detection in 5G networks," *IEEE Access*, vol. 6, pp. 7700–7712, 2018.

[151] R. K. Malaiya, D. Kwon, J. Kim, S. C. Suh, H. Kim, and I. Kim, "An empirical evaluation of deep learning for network anomaly detection," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Mar. 2018, pp. 893–898.

[152] C. Manusankar, S. Karthik, and T. Rajendran, "Intrusion detection system with packet filtering for IP spoofing," in *Proc. Int. Conf. Commun. Comput. Intell. (INCOCCI)*, Dec. 2010, pp. 563–567.

[153] N. Marir, H. Wang, G. Feng, B. Li, and M. Jia, "Distributed abnormal behavior detection approach based on deep belief network and ensemble SVM using spark," *IEEE Access*, vol. 6, pp. 59657–59671, 2018.

[154] S. Meftah, T. Rachidi, and N. Assem, "Network based intrusion detection using the UNSW-NB15 dataset," *Int. J. Comput. Digit. Syst.*, vol. 8, no. 5, pp. 478–487, 2019.

[155] W. Meng, "Intrusion detection in the era of IoT: Building trust via traffic filtering and sampling," *Computer*, vol. 51, no. 7, pp. 36–43, Jul. 2018.

[156] Y. Meng and W. Li, "Evaluation of detecting malicious nodes using Bayesian model in wireless intrusion detection," in *Proc. Int. Conf. Netw. Syst. Secur.* Cham, Switzerland: Springer, 2013, pp. 40–53.

[157] A. Meshram and C. Haas, "Anomaly detection in industrial networks using machine learning: A roadmap," in *Proc. Mach. Learn. Cyber Phys. Syst.* Cham, Switzerland: Springer, 2017, pp. 65–72.

[158] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Trans. Audio, Speech, Lang., Process.*, vol. 23, no. 3, pp. 530–539, Mar. 2015.

[159] A. R. Metke and R. L. Ekl, "Security technology for smart grid networks," *IEEE Trans. Smart Grid*, vol. 1, no. 1, pp. 99–107, Jun. 2010.

[160] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. D. Payne, "Evaluating computer intrusion detection systems: A survey of common practices," *ACM Comput. Surveys*, vol. 48, no. 1, pp. 1–41, Sep. 2015.

[161] D. Mishra. (2023). *Applications of Recurrent Neural Networks (RNNs)*. [Online]. Available: https://iq.opengenus.org/applications-of-rnn/

[162] P. Mitra, C. A. Murthy, and S. K. Pal, "Unsupervised feature selection using feature similarity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 301–312, Mar. 2002.

[163] M. Mittal, C. Iwendi, S. Khan, and A. Rehman Javed, "Analysis of security and energy efficiency for shortest route discovery in low-energy adaptive clustering hierarchy protocol using Levenberg–Marquardt neural network and gated recurrent unit for intrusion detection system," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 6, p. e3997, Jun. 2021.

[164] D. Moon, H. Im, I. Kim, and J. H. Park, "DTB-IDS: An intrusion detection system based on decision tree using behavior analysis for preventing APT attacks," *J. Supercomput.*, vol. 73, no. 7, pp. 2881–2895, Jul. 2017.

[165] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2574–2582.

[166] K. Mounika and P. V. Rao, "IDCSNet: Intrusion detection and classification system using unified gradient-boosted decision tree classifier," in *Proc. Int. Conf. Autom., Comput. Renew. Syst. (ICACRS)*, Dec. 2022, pp. 1159–1164.

[167] S. Mukkamala, G. Janoski, and A. Sung, "Intrusion detection using neural networks and support vector machines," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 2, 2002, pp. 1702–1707.

[168] S. N. Murray, B. P. Walsh, D. Kelliher, and D. T. J. O'Sullivan, "Multi-variable optimization of thermal energy efficiency retrofitting of buildings using static modelling and genetic algorithms—A case study," *Building Environ.*, vol. 75, pp. 98–107, May 2014.

[169] F. Murtagh and P. Contreras, "Methods of hierarchical clustering," 2011, *arXiv:1105.0121*.

[170] M. Musale, T. H. Austin, and M. Stamp, "Hunting for metamorphic Javascript malware," *J. Comput. Virol. Hacking Techn.*, vol. 11, no. 2, pp. 89–102, May 2015.

[171] M. N. Ahmed, A. H. Abdullah, and O. Kaiwartya, "FSM-F: Finite state machine based framework for denial of service and intrusion detection in MANET," *PLoS ONE*, vol. 11, no. 6, Jun. 2016, Art. no. e0156885.

[172] M. Nadeem, O. Marshall, S. Singh, X. Fang, and X. Yuan, "Semi-supervised deep neural network for network intrusion detection," 2016.

[173] S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, and K. Han, "Enhanced network anomaly detection based on deep neural networks," *IEEE Access*, vol. 6, pp. 48231–48246, 2018.

[174] F. Neri, "Comparing local search with respect to genetic evolution to detect intrusions in computer networks," in *Proc. Congr. Evol. Computation*, 2000, pp. 238–243.

[175] F. Nielsen, "Hierarchical clustering," in *Introduction to HPC With MPI for Data Science*. Springer, 2016, pp. 195–211.

[176] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Mach. Learn.*, vol. 39, pp. 103–134, Feb. 2000.

[177] S. Nishide, H. G. Okuno, T. Ogata, and J. Tani, "Handwriting prediction based character recognition using recurrent neural network," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2011, pp. 2549–2554.

[178] A. Nourian and S. Madnick, "A systems theoretic approach to the security threats in cyber physical systems applied to stuxnet," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 1, pp. 2–13, Jan. 2018.

[179] J. Novakovic, P. Strbac, and D. Bulatovic, "Toward optimal feature selection using ranking methods and classification algorithms," *Yugoslav J. Operations Res.*, vol. 21, no. 1, pp. 119–135, 2016.

[180] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vis. Res.*, vol. 37, no. 23, pp. 3311–3325, Dec. 1997.

[181] F. Ordóñez and D. Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, Jan. 2016.

[182] P. Hick, E. Aben, and K. Claffy. (2007). *The CAIDA DDoS Attack 2007 Dataset*. CAIDA. [Online]. Available: https://catalog.caida.org/dataset/ddos_attack_2007

[183] D. Papamartzivanos, F. Gómez Mármol, and G. Kambourakis, "Introducing deep learning self-adaptive misuse network intrusion detection systems," *IEEE Access*, vol. 7, pp. 13546–13560, 2019.

[184] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroSP)*, Mar. 2016, pp. 372–387.

[185] A. Patel, M. Taghavi, K. Bakhtiyari, and J. C. Júnior, "An intrusion detection and prevention system in cloud computing: A systematic review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 25–41, Jan. 2013.

[186] H. H. Patel and P. Prajapati, "Study and analysis of decision tree based classification algorithms," *Int. J. Comput. Sci. Eng.*, vol. 6, no. 10, pp. 74–78, Oct. 2018.

[187] A. Ponmalar and V. Dhanakoti, "An intrusion detection approach using ensemble support vector machine based chaos game optimization algorithm in big data platform," *Appl. Soft Comput.*, vol. 116, Feb. 2022, Art. no. 108295.

[188] S. Potluri, N. F. Henry, and C. Diedrich, "Evaluation of hybrid deep learning techniques for ensuring security in networked control systems," in *Proc. 22nd IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2017, pp. 1–8.

[189] M. B. Pranto, M. H. A. Ratul, M. M. Rahman, I. J. Diya, and Z.-B. Zahir, "Performance of machine learning techniques in anomaly detection with basic feature selection strategy—A network intrusion detection system," *J. Adv. Inf. Technol.*, vol. 13, no. 1, pp. 1–9, 2022.

[190] N. R. Prasad, S. Almanza-Garcia, and T. T. Lu, "Anomaly detection," *Comput., Mater., Continua*, vol. 14, pp. 1–22, Feb. 2010.

[191] T. H. Ptacek and T. N. Newsham, "Insertion, evasion, and denial of service: Eluding network intrusion detection," Secure Netw., Inc, Calgary AB, Canada, Tech. Rep., 1998.

[192] J. R. Quinlan, "Induction of decision trees," *Machine Learn.*, vol. 1, pp. 81–106, Feb. 1986.

[193] J. R. Quinlan, *C4. 5: Programs for Machine Learning*. Amsterdam, The Netherlands: Elsevier, 2014.

[194] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. 24th Int. Conf. Mach. Learn.*, Jun. 2007, pp. 759–766.

[195] P. S. Rath, N. K. Barpanda, R. Singh, and S. Panda, "A prototype multiview approach for reduction of false alarm rate in network intrusion detection system," *Int. J. Comput. Netw. Commun. Secur.*, vol. 3, no. 5, p. 49, 2017.

[196] D. Ravì, C. Wong, B. Lo, and G.-Z. Yang, "A deep learning approach to on-node sensor data analytics for mobile or wearable devices," *IEEE J. Biomed. Health Informat.*, vol. 21, no. 1, pp. 56–64, Jan. 2017.

[197] D. Ravi, C. Wong, B. Lo, and G.-Z. Yang, "Deep learning for human activity recognition: A resource efficient implementation on low-power devices," in *Proc. IEEE 13th Int. Conf. Wearable Implant. Body Sensor Netw. (BSN)*, Jun. 2016, pp. 71–76.

[198] S. Revathi and A. Malathi, "A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection," *Int. J. Eng. Res. Technol.*, vol. 2, no. 12, pp. 1848–1853, 2013.

[199] R. Roman, J. Zhou, and J. Lopez, "Applying intrusion detection systems to wireless sensor networks," in *Proc. 3rd IEEE Consum. Commun. Netw. Conf.*, 2006, pp. 1–7.

[200] S. Roy, J. Li, B.-J. Choi, and Y. Bai, "A lightweight supervised intrusion detection mechanism for IoT networks," *Future Gener. Comput. Syst.*, vol. 127, pp. 276–285, Feb. 2022.

[201] L. Rutkowski, M. Jaworski, L. Pietruczuk, and P. Duda, "Decision trees for mining data streams based on the Gaussian approximation," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 108–119, Jan. 2014.

[202] H. Sadreazami, A. Mohammadi, A. Asif, and K. N. Plataniotis, "Distributed-graph-based statistical approach for intrusion detection in cyber-physical systems," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 4, no. 1, pp. 137–147, Mar. 2018.

[203] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, Oct. 2007.

[204] R. Salakhutdinov and H. Larochelle, "Efficient learning of deep Boltzmann machines," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 693–700.

[205] M. Sarhan, S. Layeghy, and M. Portmann, "Towards a standard feature set for network intrusion detection system datasets," *Mobile Netw. Appl.*, vol. 27, pp. 1–14, Nov. 2022.

[206] S. Sarkar, K. Reddy, A. Dorgan, C. Fidopiastis, and M. Giering, "Wearable EEG-based activity recognition in PHM-related service environment via deep learning," *Int. J. Prognostics Health Manage.*, vol. 7, no. 4, pp. 1–7, Nov. 2020.

[207] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: A malicious bot-IoT traffic detection method in IoT network using machine-learning techniques," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3242–3254, Mar. 2021.

[208] M. KShah, A. M. Virparia, and K. Sharma, "An overview of advanced network steganography," *Int. J. Comput. Appl.*, vol. 118, no. 21, pp. 23–26, May 2015.

[209] S. A. Shah, D. Z. Seker, S. Hameed, and D. Draheim, "The rising role of big data analytics and IoT in disaster management: Recent advances, taxonomy and prospects," *IEEE Access*, vol. 7, pp. 54595–54614, 2019.

[210] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSP*, vol. 1, 2018, pp. 108–116.

[211] M. Sheikhan, Z. Jadidi, and A. Farrokhi, "Intrusion detection using reduced-size RNN based on feature grouping," *Neural Comput. Appl.*, vol. 21, no. 6, pp. 1185–1190, Sep. 2012.

[212] C. Shen, C. Liu, H. Tan, Z. Wang, D. Xu, and X. Su, "Hybrid-augmented device fingerprinting for intrusion detection in industrial control system networks," *IEEE Wireless Commun.*, vol. 25, no. 6, pp. 26–31, Dec. 2018.

[213] S. Shen, G. Yue, Q. Cao, and F. Yu, "A survey of game theory in wireless sensor networks security," *J. Netw.*, vol. 6, no. 3, p. 521, Mar. 2011.

[214] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, May 2012.

[215] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, Feb. 2018.

[216] M. A. Siddiqi, W. Pak, and M. A. Siddiqi, "A study on the psychology of social engineering-based cyberattacks and existing countermeasures," *Appl. Sci.*, vol. 12, no. 12, p. 6042, Jun. 2022.

[217] K. Siddique, Z. Akhtar, H.-G. Lee, W. Kim, and Y. Kim, "Toward bulk synchronous parallel-based machine learning techniques for anomaly detection in high-speed big data networks," *Symmetry*, vol. 9, no. 9, p. 197, Sep. 2017.

[218] R. Singh, H. Kumar, and R. K. Singla, "An intrusion detection system using network traffic profiling and online sequential extreme learning machine," *Expert Syst. Appl.*, vol. 42, no. 22, pp. 8609–8624, Dec. 2015.

[219] M. Soranamageswari and D. C. Meena, "Histogram based image spam detection using back propagation neural networks," in *Proc. Int. Conf. Control Automat. Syst. (ICCAS)*. IEEE, 2010, pp. 3985–3988.

[220] Y. Laarouchi, M. Kaaniche, V. Nicomette, I. Studnia, and E. Alata, "A language-based intrusion detection approach for automotive embedded networks," *Int. J. Embedded Syst.*, vol. 10, no. 1, p. 1, 2018.

[221] H. Su and C. Jung, "Perceptual enhancement of low light images based on two-step noise suppression," *IEEE Access*, vol. 6, pp. 7005–7018, 2018.

[222] S. Subramanian, V. B. Srinivasan, and C. Ramasa, "Study on classification algorithms for network intrusion systems," *J. Commun. Comput.*, vol. 9, no. 11, pp. 1242–1246, 2012.

[223] A. H. Sung and S. Mukkamala, "Identifying important features for intrusion detection using support vector machines and neural networks," in *Proc. Symp. Appl. Internet*, 2003, pp. 209–216.

[224] C. A. Sunshine, "Source routing in computer networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 7, pp. 29–33, Feb. 1977.

[225] Symantec. *Internet Security Threat Report 2017*. Apr. 22, 2017. [Online]. Available: https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf

[226] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.

[227] A. Thakkar and R. Lohiya, "Attack classification using feature selection techniques: A comparative study," *J. Ambient Intell. Humanized Comput.*, vol. 12, no. 1, pp. 1249–1266, Jan. 2021.

[228] A. Thakkar and R. Lohiya, "A survey on intrusion detection system: Feature selection, model, performance measures, application perspective, challenges, and future research directions," *Artif. Intell. Rev.*, vol. 55, no. 1, pp. 453–563, Jan. 2022.

[229] S. Thaseen and Ch. A. Kumar, "An analysis of supervised tree based classifiers for intrusion detection system," in *Proc. Int. Conf. Pattern Recognit., Informat. Mobile Eng.*, Feb. 2013, pp. 294–299.

[230] T. Verwoerd and R. Hunt, "Intrusion detection techniques and approaches," *Comput. Commun.*, vol. 25, no. 15, pp. 1356–1365, Sep. 2002.

[231] R. Thomas and D. Pavithran, "A survey of intrusion detection models based on NSL-KDD data set," in *Proc. 5th HCT Inf. Technol. Trends (ITT)*, 2018, pp. 286–291.

[232] M. Uddin, A. A. Rahman, and N. Memon, "Signature-based multi-layer distributed intrusion detection system using mobile agents," *Int. J. Netw. Secur. Appl.*, vol. 15, no. 2, pp. 97–105, 2013.

[233] T. T. Um, F. M. J. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulic, "Data augmentation of wearable sensor data for Parkinson's disease monitoring using convolutional neural networks," in *Proc. 19th ACM Int. Conf. Multimodal Interact.*, Nov. 2017, pp. 216–220.

[234] K. Vasilomanolakis, E. Vasilomanolakis, and S. karuppayah, "Taxonomy and survey of collaborative intrusion detection," *ACM Comput. Surv.*, vol. 47, no. 4, pp. 1–55, 2015.

[235] J. M. Vidal, J. M. Castro, A. S. Orozco, and L. G. Villalba, "Evolutions of evasion techniques against network intrusion detection systems," in *Proc. Int. Conf. Inf. Technol.*, 2013, pp. 1–6.

[236] J. Viinikka, H. Debar, L. Mé, A. Lehikoinen, and M. Tarvainen, "Processing intrusion detection alert aggregates with time series modeling," *Inf. Fusion*, vol. 10, no. 4, pp. 312–324, 2009.

[237] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019.

[238] R. Vinayakumar, P. Poornachandran, and K. Soman, "Scalable framework for cyber threat situational awareness based on domain name systems data analysis," in *Big Data in Engineering Applications*. Cham, Switzerland: Springer, 2018, pp. 113–142.

[239] I. Vojinovic. (2023). *Data Breach Statistics That Will Make You Think Twice Before Filling Out an Online Form*. [Online]. Available: https://dataprot.net/statistics/data-breach-statistics/

[240] C. Vollmer, H. M. Gross, and J. P. Eggert, "Learning features for activity recognition with shift-invariant sparse coding," in *Proc. Int. Conf. Artif. Neural Netw.* Cham, Switzerland: Springer, 2013, pp. 367–374.

[241] N. Walkinshaw, R. Taylor, and J. Derrick, "Inferring extended finite state machine models from software executions," *Empirical Softw. Eng.*, vol. 21, no. 3, pp. 811–853, Jun. 2016.

[242] L. Wallgren, S. Raza, and T. Voigt, "Routing attacks and countermeasures in the RPL-based Internet of Things," *Int. J. Distrib. Sensor Netw.*, vol. 9, no. 8, Aug. 2013, Art. no. 794326.

[243] A. H. Wang, "Detecting spam bots in online social networking sites: A machine learning approach," in *Proc. 24th Annu. IFIP WG Working Conf.* Rome, Italy: Springer, Jul. 2010, pp. 335–342.

[244] G. Wang, J. Hao, J. Ma, and L. Huang, "A new approach to intrusion detection using artificial neural networks and fuzzy clustering," *Exp. Syst. Appl.*, vol. 37, no. 9, pp. 6225–6232, Sep. 2010.

[245] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 869–904, 2nd Quart., 2020.

[246] Z. Wang and B. Fang, "Application of combined kernel function artificial intelligence algorithm in mobile communication network security authentication mechanism," *J. Supercomput.*, vol. 75, no. 9, p. 5843, Mar. 2019.

[247] P. Wei, Y. Li, Z. Zhang, T. Hu, Z. Li, and D. Liu, "An optimization method for intrusion detection classification model based on deep belief network," *IEEE Access*, vol. 7, pp. 87593–87605, 2019.

[248] J. Whelan, A. Almehmadi, and K. El-Khatib, "Artificial intelligence for intrusion detection systems in unmanned aerial vehicles," *Comput. Electr. Eng.*, vol. 99, Apr. 2022, Art. no. 107784.

[249] C. Wiens. (2021). *The Top 5 Zero-Day Attacks of the 21st Century*. [Online]. Available: https://securityboulevard.com/2021/07/the-top-5-zero-day-attacks-of-the-21st-century/

[250] J. Wise. (2023). *How to Bypass Firewalls in 2023*. [Online]. Available: https://earthweb.com/how-to-bypass-a-firewall/

[251] S. Wiyono and T. Abidin, "Comparative study of machine learning KNN, SVM, and decision tree algorithm to predict student's performance," *Int. J. Res. Granthaalayah*, vol. 7, no. 1, pp. 190–196, Jan. 2019.

[252] D. Wu, Z. Wang, Y. Chen, and H. Zhao, "Mixed-kernel based weighted extreme learning machine for inertial sensor based human activity recognition with imbalanced dataset," *Neurocomputing*, vol. 190, pp. 35–49, May 2016.

[253] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning: How do IoT devices use AI to enhance security?" *IEEE Signal Process. Mag.*, vol. 35, no. 5, pp. 41–49, Sep. 2018.

[254] C. Xu, J. Shen, X. Du, and F. Zhang, "An intrusion detection system using a deep neural network with gated recurrent units," *IEEE Access*, vol. 6, pp. 48697–48707, 2018.

[255] A. S. Yadav. (2021). *Top 10 Firewall/IDS Evasion Techniques*. [Online]. Available: https://medium.com/@IamLucif3r/top-10-firewall-ids-evasion-techniques-cb1e1cc06f24

[256] B. Yan and G. Han, "Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system," *IEEE Access*, vol. 6, pp. 41238–41248, 2018.

[257] Y. Yang, K. Zheng, B. Wu, Y. Yang, and X. Wang, "Network intrusion detection based on supervised adversarial variational auto-encoder with regularization," *IEEE Access*, vol. 8, pp. 42169–42184, 2020.

[258] H. Yao, D. Fu, P. Zhang, M. Li, and Y. Liu, "MSML: A novel multilevel semi-supervised machine learning framework for intrusion detection system," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1949–1959, Apr. 2019.

[259] N. Ye, S. M. Emran, Q. Chen, and S. Vilbert, "Multivariate statistical analysis of audit trails for host-based intrusion detection," *IEEE Trans. Comput.*, vol. 51, no. 7, pp. 810–820, Jul. 2002.

[260] Y. Xiao, C. Xing, T. Zhang, and Z. Zhao, "An intrusion detection model based on feature reduction and convolutional neural networks," *IEEE Access*, vol. 7, pp. 42210–42219, 2019.

[261] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[262] Y. Yuan, G. Kaklamanos, and D. Hogrefe, "A novel semi-supervised AdaBoost technique for network anomaly detection," in *Proc. 19th ACM Int. Conf. Model., Anal. Simul. Wireless Mobile Syst.*, Nov. 2016, pp. 111–114.

[263] J. Zhang, M. Zulkernine, and A. Haque, "Random-forests-based network intrusion detection systems," *IEEE Trans. Syst., Man, Cybern., C Appl. Rev.*, vol. 38, no. 5, pp. 649–659, Sep. 2008.
[264] W. Zhang, X. Lu, Y. Gu, Y. Liu, X. Meng, and J. Li, "A robust iris segmentation scheme based on improved U-Net," *IEEE Access*, vol. 7, pp. 85082–85089, 2019.
[265] X. Zhang, J. Chen, Y. Zhou, L. Han, and J. Lin, "A multiple-layer representation learning model for network-based attack detection," *IEEE Access*, vol. 7, pp. 91992–92008, 2019.
[266] X. Zhang and S. Jiang, "A splitting criteria based on similarity in decision tree learning," *J. Softw.*, vol. 7, no. 8, pp. 1775–1782, Aug. 2012.
[267] Y. Zhang, X. Chen, D. Guo, M. Song, Y. Teng, and X. Wang, "PCCN: Parallel cross convolutional neural network for abnormal network traffic flows detection in multi-class imbalanced network traffic flows," *IEEE Access*, vol. 7, pp. 119904–119916, 2019.
[268] Z. Zhang, H. Shen, and Y. Sang, "An observation-centric analysis on the modeling of anomaly-based intrusion detection," *Int. J. Netw. Secur.*, vol. 4, no. 3, pp. 292–305, 2007.
[269] G. Zhao, C. Zhang, and L. Zheng, "Intrusion detection using deep belief network and probabilistic neural network," in *Proc. IEEE Int. Conf. Comput. Sci. Eng. (CSE) IEEE Int. Conf. Embedded Ubiquitous Comput. (EUC)*, vol. 1, Jul. 2017, pp. 639–642.
[270] S. Ziweritin, B. B. Baridam, and U. A. Okengwu, "A comparative analysis of neural network and decision tree model for detecting result anomalies," *OALib*, vol. 9, no. 3, pp. 1–15, 2022.
[271] S. Zwane, P. Tarwireyi, and M. Adigun, "Ensemble learning approach for flow-based intrusion detection system," in *Proc. IEEE AFRICON*, Sep. 2019, pp. 1–8.
[272] L. Talavera, "An evaluation of filter and wrapper methods for feature selection in categorical clustering," in *Advances in Intelligent Data Analysis VI: 6th International Symposium on Intelligent Data Analysis, IDA 2005, Madrid, Spain, September 8–10, 2005. Proceedings 6*. Springer, 2005, pp. 440—451.

ZAHEDI AZAM received the B.Sc. degree in electronics and communication engineering from Khulna University, Khulna. He is currently pursuing the M.Sc. degree with United International University (UIU), Dhaka, Bangladesh. He is currently a Lead Engineer (CS and PS/IP) in the core network with Banglalink Digital Communication Ltd., Bangladesh, a leading mobile operator. His research interests include cyber security, machine learning, and deep learning. He previously work on CS, PS, and datacom telecom networks by incorporating him in patch development recommendations for telecom standards.

MD. MOTAHARUL ISLAM received the Ph.D. degree in computer engineering from Kyung Hee University, South Koreaa, in 2013. He has been a Professor with the Department of Computer Science and Engineering, United International University (UIU), Dhaka, Bangladesh, since 2020. Before joining UIU, he served many other universities at home and abroad, such as the Islamic University of Madinah, Saudi Arabia, the Islamic University of Technology, Brac University, and the University Grants Commission of Bangladesh. He has published around 100 articles in the last ten years. His research interests include the smart Internet of Things, IP-based wireless sensor networks (IP-WSNs), WSN virtualization, cloud computing, and green computing.

MOHAMMAD NURUL HUDA received the Ph.D. degree in electronics and information engineering from the Toyohashi University of Technology, Japan, in 2009. He has been a Professor with the Department of Computer Science and Engineering, United International University (UIU), Dhaka, Bangladesh, since 2013. Before joining UIU, he served many other universities at home and abroad, such as AUST, the Toyohashi University of Technology, and the University Grants Commission of Bangladesh. He has published more than a 100 articles in the last two decades. His research interests include machine translation, speech synthesis and recognition, and Bangla computing.