IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 25 July 2023, accepted 13 August 2023, date of publication 21 August 2023, date of current version 11 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3307018

RESEARCH ARTICLE

A Fine-Grained System Driven of Attacks Over Several New Representation Techniques Using Machine Learning

MOHAMMED A. AL GHAMDI
Computer Science Department, College of Computers and Information Systems, Umm Al-Qura University, Mecca 24382, Saudi Arabia
e-mail: maeghamdi@uqu.edu.sa

The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number: IFP22UQU4250002DSR228.

ABSTRACT Machine Learning (ML) techniques, especially deep learning, are crucial to many contemporary real world systems that use Computational Intelligence (CI) as their core technology, including self-deriving vehicles, assisting machines, and biometric authentication systems. We encounter a lot of attacks these days. Drive-by-download is used to covertly download websites when we view them, and emails we receive often have malicious attachments. The affected hosts and networks sustain significant harm as a result of the infection. Therefore, identifying malware is crucial. Recent attacks, however, is designed to evade detection using Intrusion Detection System (IDS). It is essential to create fresh signatures as soon as new malware is found in order to stop this issue. Using a variety of cutting-edge representation methodologies, we develop attack taxonomy and examine it. 1) N-gram-based representation: In this tactic, we look at a number of random representations that consider a technique of sampling the properties of the graph. 2) Signature-based representation: This technique uses the idea of invariant representation of the graph, which is based on spectral graph theory. One of the main causes is that a ML system setup is rely on a number of variables, including the input dataset, ML architecture, attack creation process, and defense strategy. To find any hostile attacks in the network system, we employ IDS with Deep Neural Network (DNN). In conclusion, the efficacy and efficiency of the suggested framework with Convolutional Neural Network (CNN) and Support Vector Machine (SVM) are assessed using the assessment indicators, including throughput, latency rate, accuracy and precision. The findings of the suggested model with a detection rate of 93%, 14%, 95.63% and 95% in terms of throughput, latency rate, accuracy and precision, which is based on adversarial assault, were better and more effective than CNN and SVM models. Additionally at the end we contrast the performance of the suggested model with that of earlier research that makes use of the same dataset, NSL-KDD, as we do in our scenario.

INDEX TERMS Machine learning, computational intelligence, intrusion detection system, deep neural network, convolutional neural network, support vector machine.

I. INTRODUCTION

The quantity and diversity of apps that operate on the Internet and via business IP networks have dramatically increased during the last several years. Interactive applications (like telnet, instant messaging, gaming, etc.) [1], [2], mass data transfers (like ftp, peer-to-peer file downloads) [3], [4], business applications (like lotus notes, database operations) [5], [6], and real time apps (like audio and video streaming, etc.) [7], [8] are just a few examples of the spectrum. Social media has exploded as a dominant opportunity for everyone to voice their thoughts on a variety of themes, comprising political, financial, educational, sporting, defensive, religious, and other societal concerns, owing

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

to the internet's and digital technology's spectacular rise. Despite advancements in communication technology, several challenges still exist that are essentially the key components of data transfer throughout the network. The network's information flow is plagued by the most significant issues and challenges. Depending on the network protocols and data transmission techniques chosen, the information supplied from the source travels via a number of different channels. Given the volume of data and the congestion on network pathways, whether wired or wireless, the data travels along a different route but still ends up where they started. Statistics demonstrate the exponential development of social media use, with Twitter alone producing 8K posts per second and more than 400 billion posts annually [9], [10]. The variety in use of languages all over worldwide also poses a big problem because of the wide diversity of patterns of speech. The basic aim of social media is to link more individuals in direction to facilitate their exercise of their First Amendment right to free expression. However, other organizations frequently abuse these platforms to promote insulting and hateful messages that are directed at specific people or groups in general. This might be regarded as political violence endangering societal harmony and peace. Therefore, it is crucial to identify them at the right moment and stop them from spreading to a wider group in order to maintain social peace and the status quo of law and order. Unapproved or shady computer or network activity is discovered and reported using intrusion detection systems (IDSs) [11], [12] using machine learning techniques (ML) [13], [14]. Whereas client-based IDSs, the focus of this research, are designed to monitor client system operations, network-based IDSs are used to analyze network traffic across multiple clients. IDSs [15], [16] can alternatively be categorized as abuse detection or anomaly detection based on their detection methods, contingent upon whether or not the intrusion patterns are known before the strategy process. Misappropriation exposure systems are capable to attain a high amount of detection accuracy because they search for predetermined patterns or signatures related to known assaults. However, abuse detection algorithms are unable to identify known attacks that are capable of changing their signs with each accomplishment (polymorphic attacks) [17], [18] or unknown assaults for which signs have not yet been recovered (zero-day attacks).

Modern real-world apps that engage computational intelligence (CI) [19], [20] as their crucial expertise, such as self-directed cars, assistive machines, and biometric systems, comprehensively depend on machine learning (ML) [21], [22] approaches, specially deep learning. Attacks that outcome in incorrect classifications or predictions might result in wrong decisions and unreliable activities. A significant objective in the field of adversarial machine learning is the design of robust ML [23], [24], which can deliver accurate results in the presence of such assaults. An arms race between adverse and defense organizers is a crucial factor in the quick development of strong ML. However, having accessibility to a detailed system model is necessary for independent researchers to repeat studies, which is a necessary condition for the arms race. This article suggests a fine grained system-driven classification to evidently describe adversarial system models and ML [9], [25] apps so that self-governing scholars can repeat trials and strengthen the competition to produce further sophisticated and consistent ML applications. Typically, trained datasets are gathered over a epoch of typical system action and used by anomaly detection algorithms to build profiles of expected normal behavior. To guarantee that the variance sensor is proficient on attack free data, these datasets are gathered, examined, and sanitized in a secure environment. The anomaly detection system looks for occurrences during operation that drastically differ from the predicted normal profile. These abnormalities are regarded as abnormal occurrences and are reported as such; however, they are not always the result of hostile activity since they might also be the result of software flaws (such as coding or configuration problems). Although anomaly detection techniques can identify new assaults, they frequently produce many false alarms, mostly because it is challenging to provide a fair description of the system's typical behavior. Because the base rate of regular actions outweighs the irregular ones, the anomaly detectors would consequently produce an unnecessary number of false alarms (by misclassifying infrequent normal occurrences as anomalous). This might damage the abnormality exposure system's trustworthiness. At the end of the article, we contrast the effectiveness of the suggested model with earlier research that makes use of the same dataset, NSL-KDD, which is what we do in our scenario. The key difference between our methods and those now in use is that we employ a number of cutting-edge representation approaches as well as create and analyze assault taxonomy. 1) N-gram-based representation: In this strategy, we examine several random representations that incorporate a method of sampling the graph's attributes. 2) Signature-based representation: This method, which is based on spectral graph theory, makes use of the concept of invariant representation of the graph. If we go through the current approaches, we didn't discover such a notion. This idea distinguishes our suggested paradigm from previous approaches.

II. CONTRIBUTION

The current research track suggests that deep neural networks could offer a more efficient way to construct IDS over the networks. This article proposes taxonomy of attacks and explores this taxonomy over several new representation techniques. 1) N-gram-based representation: in this approach we explore various random representations that would consider a sampling technique over the graph properties (e.g., random walk based). 2) Signature-based representation: this technique exploits the idea of invariant representation of the graph based on the spectral graph theory. One of the main causes is that an ML application's setup depends on the 1) input dataset, 2) ML architecture, 3) adversary requirements, 4) attack generating process and defense strategy, among

other things. It may be highly challenging to repeat trials on the ML application when one or more of these elements, each of which includes several smaller components, are unknown or unidentified. In terms of the aforementioned characteristics, this work offers a fine grained system-driven classification of ML apps in adversarial environments. These taxonomies may be utilized to openly define each component of the ML systems, allowing for the unmistakable repetition of the trial, encouraging a research arms race, and perhaps fostering the development of future ML applications that are more reliable. Additionally, this taxonomy might help in assessing novel study in respect of experiment repeatability and influence on full-bodied adversarial ML strategy.

III. RELATED WORKS

Previous research in this section looked at how flow properties varied depending on application. Attacks are a widespread occurrence on the Internet nowadays, and despite intensive study over the past few years, no universally applicable preventative methods have yet been developed. One factor is the adversaries rapid spread, which gives the defense very little time to respond. Another challenge is the problem's dispersed character, which necessitates the deployment of defense systems practically gapless across the Internet. A novel semi-self-taught (SST) network intrusion detection system based on a semi-supervised discriminant auto encoder (SSDA) is introduced by the authors in [26], and it only needs a little amount of human input. A demonizing auto encoder using a k-mean technique was used to build SSDA for class recognition. The CSE-CICIDS2018 dataset is the foundation of the experiment. Because there is just one dataset, this might lead to an overfitting issue. This approach also falls short in terms of data fusion. In [27], authors provide a brand-new XGBoost deep neural network (DNN) framework for network intrusion exposure. First, data preparation is carried out in order to normalize the data. Second, the XGBoost approach is applied to high-dimensional data feature selection. Finally, a DNN technique is employed for binary taxonomy. During model training, the Ant optimizer (AO) is used to boost the learning rate. The NSL-KDD dataset is used in the trials. The single dataset of this approach contributes to its overfitting issue. Additionally absent in this process are model assessment and data fusion. Through the use of an algorithm driven method, Ling et al. [28] create a classification by offering a platform that employs a number of cutting-edge assaults and defenses. The platform allows for the testing and evaluation of deep learning apps in response to assaults and the efficacy of various defenses.

The issue with string fingerprints in the context of polymorphic worms was initially brought up by Newsome et al. [29]. Their ''Polygraph'' method suggests taking numerous invariant byte sequences that are shared by all interpretations of a synthetic polymorphic attack. The researchers demonstrate how specific adjoining byte sequences, including protocol edging series and the high demand bytes of buffer excess reoccurrence addresses, typically persist consistent through every single instance of a polymorphic attack and may so be utilized to create a worm signature. A taxonomy focused on the impact of adversary information on the attack accomplishment frequency and potential defense measures in an online PDF malware detector is proposed by Laskov [30] as an illustration of an application-driven taxonomy. These studies offer a thorough examination of the weapons race, although they are heavily reliant on certain data types, techniques, or applications. They might not be able to effectively adjust in this situation to handle a variety of applications or possible dangers in the future. When new malware is identified, Choi et al. [31] provided a way to create a hierarchical signature cluster tree from the current network signatures and a plan to create new signatures quickly by associating with the hierarchical signature cluster tree. In [1], employing AE incorporating CNN, two fully connected layers, and the output to the softmax classifier, multi-channel deep learning of features for NIDS was described. With an average accuracy of 94%, the examination is conducted over three separate datasets: KDD cup99, UNSW-NB15, and CICIDS. The findings produced by the suggested model are efficient, but the attack's structure and traits weren't made abundantly evident.

A deep learning based intrusion exposure method for 5G networks was suggested by Maimó et al. [32]. Their suggested IDS has two level: low-level for detecting abnormal symptoms and high-level for detecting network anomalies. Deep Neural Network and Long-Short Term Memory (LSTM) network were the deep learning methodologies utilized for low-level and high-level, respectively. The network behavior and computing resources are automatically adjusted by these two levels of IDS. The entire project was carried out using CTU dataset, which was based on botnets. Recurrent Neural Network (RNN)-based new IDS was proposed by Yin et al. [33]. For creating IDS, they used the NSL-KDD dataset. Additionally, data preparation has been done to assess an IDS's effectiveness. Binary classification and multi-classification are the two difficulties that their suggested study falls under. On the test dataset, the suggested model achieves 97.09% accuracy. They evaluated how well the RNNIDS model performed in comparison to other machine learning techniques already in use. The authors continue to focus on employing GPU acceleration to shorten training times, avoiding explosion and disappearing rises, and analyzing the taxonomy efficacy of the LSTM and bidirectional RNNs algorithms in the intrusion detection space. Extreme gradient-boosting (XGBoost) classifiers were employed in [34] to discriminate between regular and DoS assaults. The analysis and testing of the detection approach was done using POX SDN as a controller, an open-source SDN podium for designing and prototyping SDN based methods. To mimic real-time SDN-based cloud identification, the network architecture was simulated using Mininet. The learning technique chosen was logistic regression, with a regularization term drawback to avoid overfitting.

The logistic regression technique was integrated with the XGBoost term to improve calculations by building structure trees. KDD Cup 1999 was the dataset utilized in this methodology, and 500 K samples were chosen to build the training set. Both a logarithmic based approach and a Min-Max-based strategy were employed as different sorts of normalization procedures. Comparing XGBoost to RF and SVM, the average overall accuracy was 98%, 96%, and 97%, respectively.

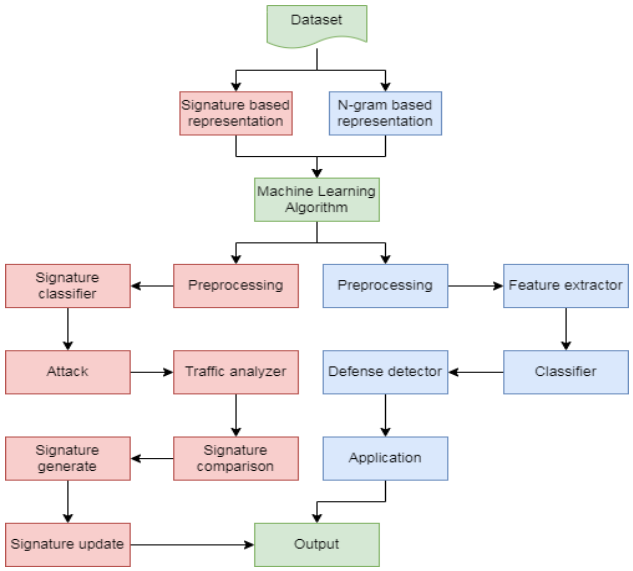

FIGURE 1. The framework of the proposed intrusion detection model.

IV. THE PRPOSED METHOD

This section offers attack taxonomy and investigates this taxonomy using a number of novel representation strategies on same dataset as shown in Fig.1: 1) N-gram-based representation: In this strategy, we investigate several random representations that would consider a sampling method across the attributes of the graph (for example, random walk based). 2) Signature-based representation: This method makes use of the spectral graph theory-based concept of invariant representation of the graph. One of the key explanations is that the formation of an ML system depends on a variety of aspects, including the 1) input dataset, 2) ML architecture, 3) attack generation process, and 4) defense strategy. Repeating trials on the ML application may be extremely difficult if one or more of these parts, each of which has a number of smaller components, are unknown or unidentified. This paper provides a fine grained system-driven classification of ML systems in hostile contexts based on the aforementioned criteria.

A. N-GRAM-BASED REPRESENTATION

The formal descriptions of both word vector weighting schemes are presented first in this section. The suggested method for effective n-gram feature extraction based on variable length is discussed. Assume that $G = a_1, a_2, \ldots, a_N$ is a suggestion of scheme call interpretations (ai) of length L, created by a practice with an alphabet of size $m = |\Sigma|$ (unique) system calls. $G = G_1, \ldots, G_K$ denotes the crew of K suggestions produced by the procedure (or system) of attention and then delivered for developing the intrusion detection framework. Every trace $V \epsilon V$ in the binary phrase vector, Ó(V), is translated into a route of mass n system requests, $V \rightarrow Ó(V)_{Ó} \epsilon \Sigma$, where every time or function $o_i \epsilon \Sigma$ is given a binary flag based on whether it appears in the trace V (one) or not (zero). By using the time frequency (tf), the term vector may be weighted:

$$Ó_{tf}(o, V) = freq(o_i); i = 1 \ldots .n \quad (1)$$

where M (the total number of systems calls in V) is multiplied by freq (the frequency of a system call occurring in V). All terms are treated equally relevant throughout all documents or collections of traces (V) when using the term frequency. In contrast to phrases that are common in most papers, uncommon terms that regularly exist in a limited number of forms deliver further statistics. It is suggested that phrases that are uncommon or common across all documents have their weights increased or decreased using the inverse document frequency (idt). Therefore, the following is the definition of vector weighted by the tf.idt:

$$Ó_{tf.idt}(o, V, \tau) = \frac{K}{dt(o_i)} fre(o_i); i = 1 \ldots .n \quad (2)$$

where the number of traces V_k in the crew V of size K that comprises system request oi is the document frequency dT (o_i). Thus, system requests that happen often in a specific trace $V\epsilon\ \tau$ nevertheless in limited or no other traces in the crew V are assigned a high weight in tf.idt. Eq. (2) has a number of variations that, for example, take the opposite document frequency logarithm or use alternative normalization factors. However, both weighting algorithms ignore the temporal sequence of system calls, as illustrated in Eqs. (1) and (2). However, the suggested method takes into consideration the time-based instruction of system calls by take-out variable length n-grams and their occurrences from every trace $V\epsilon V$ and mapping them to fixed size feature vectors. Each n-gram is a collection of consecutive system requests of length L that have been taken from trace V.

B. SIGNATURE-BASED REPRESENTATION

Two objectives drive this section. The first step is to group signatures into categories. The second is to create new signatures quickly and simply by comparing them to the ones that already exist. We describe in what way to cluster IDS regulations (such as signatures) in this section. A signature classifier in Fig.1 creates a signature cluster tree with a hierarchical arrangement when it receives signatures, as seen in Fig. 1. The following Eq. (3) is the produced signature cluster tree. IDS systems employ leaf nodes as signatures. The longest common substring (LCS) method is used to compare different signatures. The next step is to cluster the

two signatures with the top resemblance score, and an LCS-valued cluster node is created as a result. The similarity score SSx,y of two signatures is calculated as follows when their longest common substrings are LCSx,y and Sx and Sy, respectively.

$$SS_{x,y} = len\left(LCS_{x,y}\right) / \left(len\left(S_x\right) + len\left(S_y\right)\right) \tag{3}$$

It is then contrasted with additional signature nodes or cluster nodes. A cluster node is formed by grouping together two more nodes with the next highest similarity score. Until there are no more nodes with any shared substrings greater than 7 bytes, these procedures are repeated. Finally, in Fig.1 creates a signature cluster tree. In the second step of this section, a fresh malware sample is analyzed, and a novel signature is created by contrasting it to a hierarchical signature cluster tree. As seen in Fig. 1, a fresh signature is created. Network packets produced by the new malware are gathered when it is launched, and we use the network packet analysis to create a temporary signature. In the end, a new signature is formed by contrasting the short-term signature to the hierarchical signature cluster tree.

C. ML ARCHITECTURE

This stage involves building the DNN classifier. A feed-forward neural network, a successor of the traditional artificial neural network, is the DNN. The input, hidden, and output layers make up the three levels of DNN. The preprocessed input data are fed into the network by the input layer. The dataset's input characteristics to the neural network have the same number of input neurons. The input layer with N inputs is therefore represented as in Eq. (4).

$$X = [x_1, x_2, x_3, \ldots x_N] \tag{4}$$

The deep learning network allows for the addition of several hidden layers; hence the next layer is the hidden layer. With arbitrary weights (wi) and a bias (bj), the hidden layer translates the input X from the input layer. As a result, inputs to hidden layers are represented as in Eq. (5).

$$h_k = \sum\nolimits_i w_i x_i + b_k \tag{5}$$

where k = 1, 2, 3…j is the number of hidden elements in the DNN. There is a nonlinear activation function linked to each hidden layer. Better performance and faster training times for deep neural networks have been made possible by ReLU. The elimination of the disappearing and expanding gradient issue was ReLU's primary innovation. Despite having strong nonlinear and non-differentiate at zero, rectifier neurons perform better than sigmoid and hyperbolic refraction neurons. As a result, using genuine zeros to create sparse representations yields impressive results that are appropriate for sparse data. For 100 epochs of binary classification, sigmoid and tanh activation functions both exhibit respectable exposure rates compared to ReLU. When the experiment is run for 500 epochs, ReLU outperforms sigmoid and tanh activation functions in terms of performance. As a result, we made the decision to use ReLU activation functions to build our suggested framework. The outcomes of the hidden layer are presented as follows in Eq. (6).

$$h = f\left(h_k\right) \tag{6}$$

where in Eq. (7),

$$f\left(h_k\right) = ReLU\left(h_k\right) \tag{7}$$

The activation function of the outcome layer receives inputs from the hidden layer and generates the DNN's outputs. For categorization of intrusion detection inputs into a class of possibilities called $£(X)_k$, we employed the non-linear activation utility softmax in the outcome layer. The DNN's output is represented as in Eq. (8).

$$£(X)_k = \frac{e_k^x}{\sum_{j=1}^{j} e_j^x} \tag{8}$$

where k catalogs the amount of outcome elements and X is a vector of inputs to the outcome layer. In this DNN configuration, network training is done utilizing the participations to each class outcome (for example, normal 1 and abnormal 0). The weight of each input link is changed repeatedly to reduce training mistakes when the DNN is trained using a sizable training dataset. The model parameters of the DNN are modified to train the network more quickly and effectively. When the learning algorithm is being trained, these tuning parameters, or hyper parameters, are employed to regulate optimization processes and model choice. These hyper parameters determine whether the model is under fitting or over fitting throughout the training process. We present a novel taxonomy for network intrusion detection model using DNN, which primarily incorporates two additional module, feature abstraction and encoder, as shown in Fig.2, in order to increase the accuracy of adversary exposure in binary-classification and multi-classification, considering the aforementioned variables. The latent layer input is compressed using DNN, but the features will still be redundant. The attention mechanism is then utilized to learn the latent layer features. As of now, we have finished extracting the essential components from network data using an encoder. The encoder's latent layer is then linked to a DNN and categorized using the softmax function. The NSL-KDD dataset is thoroughly evaluated to determine the efficacy of the DNN model, and the best results are attained.

1) DATASET DESCRIPTION

Numerous malware detection studies make use of the 2009-created NSL-KDD dataset [21], [22]. In the most recent literature [23], [24], [25], every researcher uses the NSL–KDD standard dataset that not only resolves the intrinsic redundant accounts issues of the KDD Cup 1999 dataset in an efficient manner but also maintains an appropriate number of accounts in the training set and testing set so that the classifier does not approval more recurrent accounts. The KDDTrain+

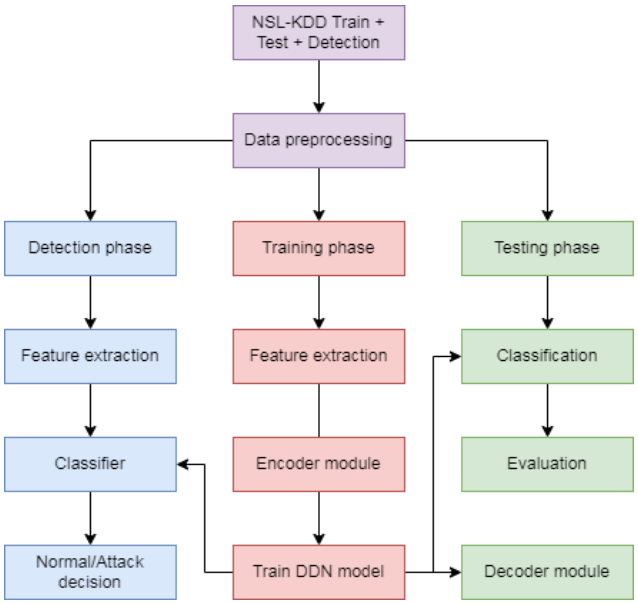

FIGURE 2. A generic system model of intrusion detection machine learning cycle.

TABLE 1. Comparison with existing studies.

Ref.	ML techniques	Data fusion	Attack diversity	Tool	Evaluation metrics	Dataset
[26]	Semi-Self-Taught	×	Yes	Anaconda Keras	×	CSE-CIC-IDS2018
[27]	CNN	×	No	Keras	✓	NSL-KDD
[28]	DL	✓	Yes	N/A	✓	MNIST & CIFAR-10
[29]	N/A	×	No	N/A	×	N/A
[30]	SVM	✓	No	N/A	✓	Surrogate
[31]	N/A	×	Yes	N/A	×	N/A
[1]	DL	✓	Yes	Python	✓	KDDCUP99
[32]	DL	×	No	N/A	×	CTU
[33]	RNN	×	Yes	N/A	✓	NSL-KDD
[34]	SVM	×	Yes	Mininet	✓	N/A
This work	Graph CNN	✓	Yes	Anaconda Keras		NSL-KDD

dataset is used as the training set, while the KDDTest+ and KDDTest-21 datasets are used as the testing set. These datasets contain various regular accounts and five distinct kinds of attack accounts. A more challenging dataset for classification is the KDDTest-21 dataset, which is a subset of the KDDTest+. As indicated in Table 2, each traffic account includes a single class label and 41 characteristics, which are divided into fundamental characteristics (1–10), content characteristics (11–22), and traffic characteristics (23–41). Attacks in the dataset are divided into five attack classes based on their traits: Normal, DDoS (Distributed-Denial-of-Service) attacks, MITM (Man-In-The-Middle) attacks, DNS (Domain-Name-System) attacks, and BF (Brute-Force) attacks. It is possible to give an intrusion detection theoretical foundation that seems more credible by using the testing set since it contains certain particular attack types that are absent from the training set. The NSL-KDD dataset contains of 3 nonnumeric characteristics and 38 numeric characteristics.

TABLE 2. Features for NSL-KDD dataset.

No.	Features	Types	No.	Features	Types
1	duration	Continuous	22	Is_guest_login	Symbolic
2	protocol_type	Symbolic	23	Count	Continuous
3	service	Symbolic	24	src_count	Continuous
4	flag	Symbolic	25	serror_rate	Continuous
5	scr_byte	Continuous	26	srv_serror_rate	Continuous
6	dst_byte	Continuous	27	rerror_rate	Continuous
7	land	Symbolic	28	srv_rerror_rate	Continuous
8	wrong_fragment	Continuous	29	same_srv_rate	Continuous
9	urgent	Continuous	30	diff_srv_rate	Continuous
10	hot	Continuous	31	srv_diff_host_rate	Continuous
11	num_failed_logins	Continuous	32	dst_host_count	Continuous
12	logged_in	Symbolic	33	dst_host_srv_count	Continuous
13	num_compromised	Continuous	34	dst_host_same_srv_count	Continuous
14	root_shell	Continuous	35	dst_host_diff_srv_count	Continuous
15	su_attempted	Continuous	36	dst_host_same_src_port_rate	Continuous
16	num_root	Continuous	37	dst_host_srv_diff_host_rat	Continuous
17	num_file_creations	Continuous	38	dst_host_serror_rate	Continuous
18	num_shells	Continuous	39	dst_host_srv_serror_rate	Continuous
19	num_access_files	Continuous	40	dst_host_rerror_rate	Continuous
20	num_outbound_cmds	Continuous	41	dst_host_srv_rerror_rate	Continuous
21	is_host_login	Symbolic			

We have to transform some nonnumeric characteristics, such as "protocol_type," "service," and "flag" features, into numeric method since the input rate of DNN-IDS must be a numeric value. For instance, the property "protocol_type" of the feature contains three different types of characteristics: "tcp," "udp," and "icmp," and its numerical matrix are represented as binary vectors (1,0,0," (0,1,0"), and (0,0,1), correspondingly. Related to how the characteristic "service" has 80 kinds and "flag" has 13 types, both features have attributes. Keeping up with this pattern, the transformed 41-dimensional characteristics transfer into 132-dimensional characteristics. Firstly, we employ the logarithm-based scaling process for scaling to determine the series of some characteristics, such as "duration [0,60329]," "src_bytes [0,1.3209]," and "dst_bytes [0,1.3209]," where the disparity among the highest and lowest elements takes a actual huge possibility. Secondly, every characteristic value is linearly translated to the [1 0] range according to Eq. (9), where Max and Min stand for the feature's maximum and minimum values, respectively.

$$n_i = \frac{n_i - Min}{Max - Min} \tag{9}$$

2) ATTACK GENERATION PROCESS AND DEFENSE STRATEGY

The four major characteristics of an adversary knowledge, ability, objective, and methods can be used to create an assault model as shown in Fig 3. The proportion of each sample containing non-target input that are incorrectly identified as the objective class out of all assaults is known as the adversaries Success Rate (SR). An adversary may have access to two major sorts of knowledge: data and algorithms. The adversary's knowledge may be partial or complete for each category. In contrast to an enemy with limited information, a neural network design may be described by the number

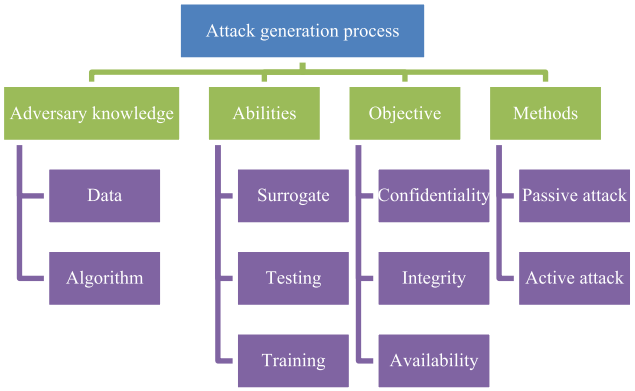

FIGURE 3. Attack generation process taxonomy.

of layers, the amount of nodes in every layer, and the kind of alteration functions. It is important to realize that the opponent might assume greater knowledge as the assault is being planned. For instance, the adversary can acquire losses in later phases if they are aware of the classifier design and settings. In real-world settings, an adversary has restricted admittance to the objective method or is only able to change a subset of the input (surrogate, testing, or training). The size of the chunk or the number of potential admittances plays important parts in the accomplishment degree of the attacker. In numerous adversarial ML studies, the adversary is believed to have complete class label understanding that is gained both by theft labeled data and by gaining oracle admittance to the ML app. While the structure safety is assessed underneath the worst case situation (from the target system's perspective) by supposing full understanding for an attacker, this presumption does not always true. Focusing on three serious security violations, the adversary's objectives may be divided into three primary groups: 1) confidentiality, 2) integrity, or 3) availability, sometimes referred to as the CIA trio. These objectives can either is: There are two types of attacks: 1) undiscriminating, where any entity information or any ML app component is vulnerable, and 2) targeted, which goals a particular entity or module. In an undiscriminating alarm assault, the foundation records are incorrectly categorized in all goal entities, but in the battered form, it must be incorrectly identified as input in a particular target entity. The two basic categories of attack methods are passive attack and aggressive attack. Deriving information about the program or its users is the goal of passive attacks. Active assaults prevent the program from functioning normally. Defense strategies are created using one of two major methodologies: either proactive, in which case a target system is prepared for promising vulnerabilities earlier an attack, or reactive, in which case security strategies are used after violence has already occurred, such as machine learning following assassinating assaults. The defense expansion appearances how much the adversary's success rate is decreased when a defense is used. For greater security improvement, numerous security tactics can potentially be integrated (identified as collaborative resistances). Adversarial samples frequently occur inside the target entity edge that frequently encompasses a large area with blind spots. By employing adversary instance thwarting, a classification system can modify its deep learning algorithm or training process to eliminate blind spots or restrict its exposure to them.

V. IMPLEMENTATION DETAILS

The suggested approach has been put into practice in Python 3.10 by means of the Keras 2.36 public library and TensorFlow 8 as the backend. There is virtual access to the source code. The research study is run on a ThinkPad Z16 personal notebook through an Intel Core i5-5200U CPU clocked at 2.20 GHz, 16 GB of RAM, and no GPU deceleration. To examine the accuracy of the DNN-IDS approach for binary classification (Normal, anomaly) and five kind classifications, including Normal, DDoS, MITM, DNS, and BF, two experiments have been developed. Concurrent comparison studies are aimed to compare with various machine learning approaches. We have examined the performance of proposed model with convolutional neural network (CNN), support vector machine (SVM) and other machine learning algorithms in the binary classification trials. In a similar vein, we examine the DNN-IDS model's multi-classification using the NSL-KDD dataset. In comparison, we investigate how well the DNN, CNN SVM and other machine learning techniques perform in the categorization of five different categories. Lastly, we evaluate the efficiency of the DNN-IDS framework in comparison to conventional approaches. Additionally, we build the dataset and evaluate performance using the reduced-size DNN approach. In the case of binary classifier trials, the DNN-IDS model contains 132 nodes for input and 2 nodes for output since we have translated 41-dimensional characteristics into 132-dimensional characteristics. There are 200 epochs in all. To develop a more effective model, set the hidden node counts at 40, 80, 120, 160, and 320, respectively, and set the learning rates at 0.01, 0.1, and 0.5, correspondingly. Next, we test the model's performance on the NSL-KDD dataset by measuring classification accuracy. The various outcomes demonstrate that the correctness is correlated with the quantity of hidden nodes and the learning degree.

VI. RESULTS AND DISCUSSION

The metrics for evaluating adversarial assaults and defensive performance were covered in this section. The Deep Neural Network (DNN) classifier, a detection model that separates between abnormal and normal using the deviation scores, is used to classify and acquire the classification results. Every sample in the NSL-KDD test dataset, the database utilized in this study, comprises 41 characteristics. Normal, DDoS (Distributed-Denial-of-Service) attacks, MITM (Man-In-The-Middle) attacks, DNS (Domain-Name-System) attacks, and BF (Brute-Force) attacks are shown in Fig4. All labels aside from the normal denote the various assaults in the dataset. Normal and assault are the two subcategories of the

NSL-KDD data set. The anomaly exposure model's key goal is to attain high accuracy with a low false alarm rate (FP).

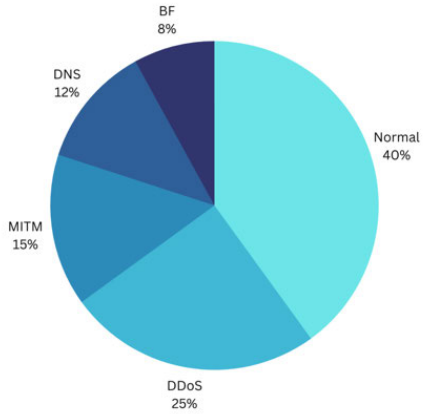

FIGURE 4. Distribution of attacks.

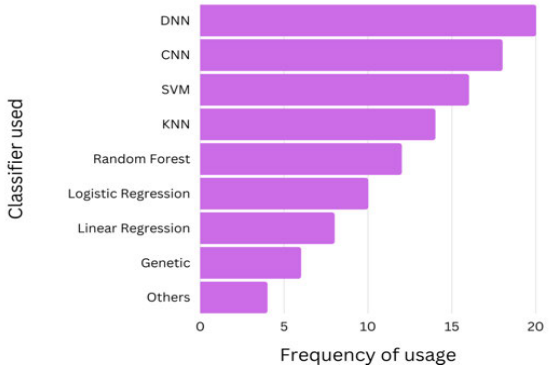

FIGURE 5. Frequency of classifiers used.

The frequency with which each classifier was applied in the related work presented in Section III is seen in Fig 5. It is obvious that the researchers are utilizing ever more deep learning methods, such CNN or SVM. Due to DNN excellent performance in spotting minor classes, it is also evident that we are utilizing it as classifier. As a result, we frequently stack it on top of other machine learning techniques. Additionally, ensemble approaches are commonly utilized since we enable the combination of many strategies to increase IDS effectiveness. The development of these more potent approaches is made feasible by advances in GPU utilization and network infrastructure. Finally, compared to CNN and DNN, older classical ML methods like K nearest neighbor, K-mean clustering, and genetic algorithms are less popular.

Fig. 6 depicts the examination of the measures employed to rate the various solutions investigated. Accuracy and throughput are the two measures that are most frequently employed. These indicators are crucial for determining a solution's level of quality. As a result, they must constantly be considered when evaluating an IDS's effectiveness. However, we believe the latency rate should also be more frequently

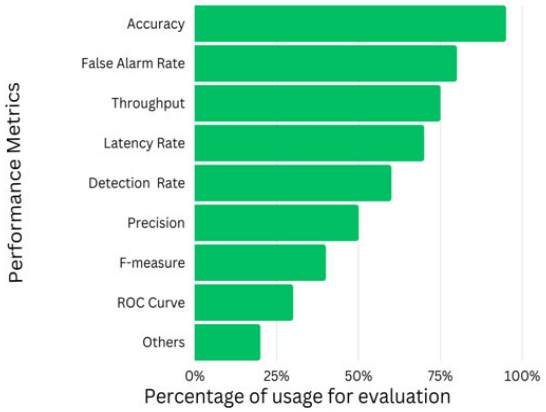

FIGURE 6. Percentage of evaluation metrics.

employed when evaluating IDS since it demonstrates if the result is effective to discover illustrations that are in fact assaults, even from negligible classes.

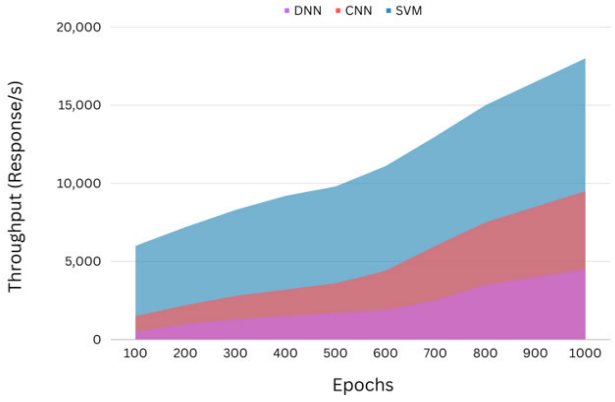

FIGURE 7. Evaluation of throughput.

A. EVALUATION OF THROUGHPUT

Fig. 7 displays the proposed system's typical reaction times for each of the three testing conditions. As we can see, both the CNN and the SVM are responsible for the overhead of the suggested system. The suggested model performs just a little bit better than the CNN and the SVM due to its reduced complexity. However, the CNN outperforms the SVM in terms of detection accuracy placements. Both the CNN and the SVM are expected and necessary. The throughput somewhat decreases when the network's epoch count rises from 200 to 1000. Performance drops by about 4.5% when the network has fewer than 1000 epochs. When the number of epochs is increased over the network, the throughput may roughly decrease.

B. EVALUATION OF LATENCY RATE

The delay rate stays constant as the network size grows, as seen in Fig. 8. Expanding the network, which is not overhead, also puts more strain on the system. The suggested

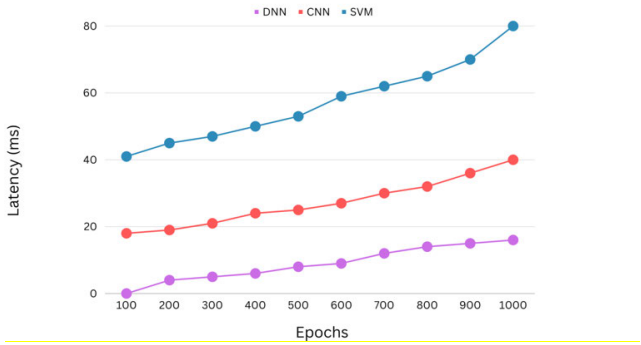

FIGURE 8. Evaluation of latency rate.

system is unaffected by the highest overhead. The total drop in Fig. 8 is close to 15%. Overall, compared to the CNN and the SVM, the suggested model overhead is rather low. As an outcome, our proposed method shows great promise for wireless systems' real-time adversarial attack recognition. All network administrators must handle this trade-off between performance and safety from the perspective of the network. Table 3 compares a classifier with other ideas that have been put forth.

TABLE 3. Classifiers comparison for feature dataset.

Classifiers	Dataset	Throughput	Latency rate	Accuracy	Precision
CNN	KDD99	75%	18%	85.02%	84%
SVM	UNIXDS	65%	20%	70.06%	70%
Proposed Model (DNN)	NSL-KDD	93%	14%	95.63%	95%

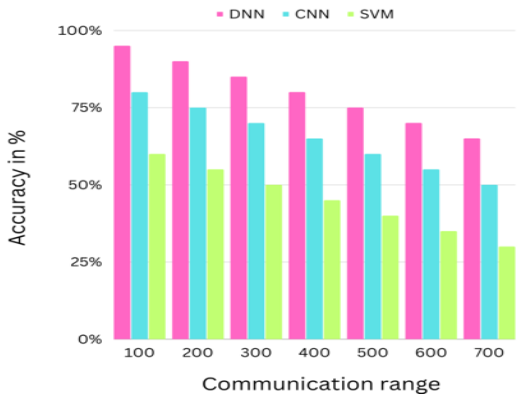

FIGURE 9. Evaluation of accuracy.

C. EVALUATION OF ACCURACY

Our results were compared to a number of different machine learning methods. Other writers use a full training and testing set of forty-one attributes to train and test several algorithms. The authors can evaluate how well these algorithms perform on their dataset based on the outcomes of these tests. Fig. 9 demonstrates that, when compared to other methods, our DNN methodology performs only moderately well, with an accuracy of 95.63%. CNN is the most precise machine learning method, with a precision of 85.02%. Since this accuracy was reached utilizing the whole feature training set, the SVM approach can generalize the regular and anomalous traffic features very effectively.

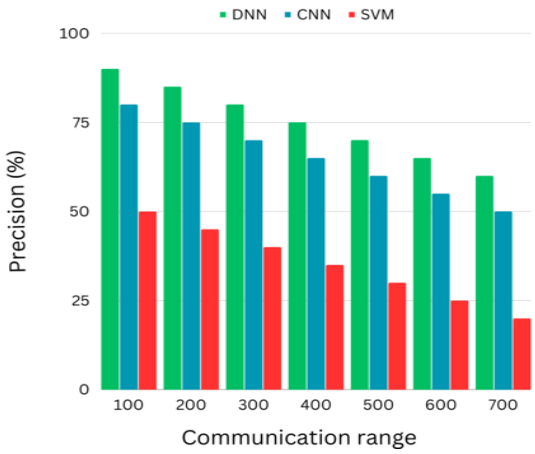

FIGURE 10. Evaluation of precision.

D. EVALUATION OF PRECISION

The suggested DNN-based technique, as shown in Fig. 10, has produced greater precision metrics for current communication ranges between 90% and 95%. However, as the best results for each prediction approach, the CNN method only reached 84% only for communication range 100 with high accuracy, and the SVM algorithm showed 70% just for communication range 200. This assessment shows that for identifying assault situations, the SVM method only has a little degree of precision below 70%. With regard to, we draw the conclusion that the accuracy factor in various algorithms has distinct assessment outcomes with different communication ranges. The suggested DNN-based technique has still achieved the highest level of precision for the total communication ranges.

E. CLASSIFIER COMPARISON FOR FEATURE DATASET

We use the feature set for training and testing in this part to conduct additional evaluation. In the subsequent experiment, the proposed DNN approach from this work was compared against machine learning methods. The results of each machine learning method on a sub-feature dataset are summarized in Table 3. Using the provided DNN algorithm, we achieve the best results from any technique. The other machine learning methods perform poorly because, with only six variables, they are unable to sufficiently generalize the traits of training samples. The feature dataset accuracy comparison is shown in Table 3. Finally, we evaluated various model accuracy using our suggested model to various evaluation matrices using already available methods and feature datasets. Compared to previous methods, the improved DNN model has a stronger influence on detection. The deep neural network algorithm using SVM performs superior to

others, as expected. Compared to earlier algorithms, our DNN technique has a greater accuracy frequency and a inferior false positive frequency. With the aid of the data shown above, we were able to demonstrate how our recommended DNN approach abstracts and generalizes the characteristics of both normal and aberrant traffic with a low number of features and a promising degree of accuracy.

TABLE 4. Performance comparison of proposed model with existing studies using same dataset.

Ref.	Classifiers	Dataset	Throughput	Latency rate	Accuracy	Precision
[27]	CNN	NSL_KDD	72%	19%	89.02%	89%
[33]	SVM	NSL_KDD	69%	24%	75.06%	74%
[35]	BAT-MC	NSL_KDD	82%	22%	87.67%	88%
[36]	DRL	NSL_KDD	79%	16%	90.23%	80%
[37]	N/A	NSL-KDD	75%	18%	86.56%	82%
[38]	LSTM	NSL-KDD	80%	25%	84.98%	70%
Proposed Model	DNN	NSL-KDD	93%	14%	95.63%	95%

F. PERFORMANCE COMPARISON OF PROPOSED MODEL WITH EXISTING STUDIES USING SAME DATASET

In this section we compare the performance of proposed model with existing studies that use same dataset NSL-KDD which we use in our scenario. In Table 4 we summarize all results by using different key metrics (e.g., Throughput, Latency Rate, Accuracy and Precision). Results shows clearly that our proposed model outperform than existing methods. To find any hostile attacks in the network system, we employ IDS with Deep Neural Network (DNN). The main thing that improves our results and different from exiting techniques is that we use a variety of cutting-edge representation methodologies; we develop attack taxonomy and examine it. 1) N-gram-based representation: In this tactic, we look at a number of random representations that consider a technique of sampling the properties of the graph. 2) Signature-based representation: This technique uses the idea of invariant representation of the graph, which is based on spectral graph theory. If we go through existing methods, we didn't find as such idea. This concept makes our proposed model different and unique from existing methods.

VII. CONCLUSION

In this study, perturbation assaults on ML applications are analyzed by creating system driven framework and classifications of the apps, opponent tactics, defenders, and their interconnections. The study works may be ranked and compared using the offered models and taxonomies to identify possible weaknesses that have not yet been addressed by the present investigations. The taxonomies may be generalized to assess and set up the upcoming study since they are self-governing from a particular kind of data, method, or app. We offer attack taxonomy and investigates this taxonomy using a number of novel representation strategies. 1) N-gram-based representation: In this strategy, we investigate several random representations that would consider a sampling method across the attributes of the graph (for example, random walk based). 2) Signature-based representation: This method makes use of the spectral graph theory-based concept of invariant representation of the graph. Its objective is to make the process of quickly and simply creating a novel signature after novel attack is found. By associating them to the hierarchical signature cluster tree, new signatures are created when malware is launched on a virtual machine without IDS raising an alarm. In conclusion, throughput, latency rate, accuracy, and precision are used as assessment indicators to calculate the accuracy and efficiency of the anticipated framework using CNN and SVM. The results of the recommended model, which is based on adversarial assault and has detection rates of 93%, 14%, 95.63%, and 95%, were superior to those of CNN and SVM frameworks in respect of throughput, latency rate, accuracy, and precision. In order to verify the accuracy of our findings, we want to carry out further tests using datasets gathered from different operating systems in future work. Additionally, we want to look at combining several machine learning methods that have been competent on the suggested feature directions with consecutive learning indicators.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number: IFP22UQU4250002DSR228.

REFERENCES

[1] 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.
[2] S. H. Javed, M. B. Ahmad, M. Asif, S. H. Almotiri, K. Masood, and M. A. A. Ghamdi, "An intelligent system to detect advanced persistent threats in industrial Internet of Things (I-IoT)," *Electronics*, vol. 11, no. 5, p. 742, Feb. 2022.
[3] Z. Chen, Q. He, B. Jiang, L. Cao, and F. Li, "HyPDS: Enabling a hybrid file transfer protocol and peer to peer content distribution system for remote sensing data," in *Proc. IEEE 25th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Dec. 2019, pp. 529–534.
[4] M. A. Darwish, E. Yafi, M. A. Al Ghamdi, and A. Almasri, "Decentralizing privacy implementation at cloud storage using blockchain-based hybrid algorithm," *Arabian J. Sci. Eng.*, vol. 45, no. 4, pp. 3369–3378, Apr. 2020.
[5] É. R. Keresztes, I. Kovács, A. Horváth, and K. Zimányi, "Exploratory analysis of blockchain platforms in supply chain management," *Economies*, vol. 10, no. 9, p. 206, Aug. 2022.
[6] A. Ahmad, M. Saad, M. Al Ghamdi, D. Nyang, and D. Mohaisen, "BlockTrail: A service for secure and transparent blockchain-driven audit trails," *IEEE Syst. J.*, vol. 16, no. 1, pp. 1367–1378, Mar. 2022.
[7] I. M. I. Zebari, S. R. M. Zeebaree, and H. M. Yasin, "Real time video streaming from multi-source using client-server for video distribution," in *Proc. 4th Sci. Int. Conf. Najaf (SICN)*, Al-Najaf, Iraq, Apr. 2019, pp. 109–114.
[8] M. A. A. Ghamdi, "An optimized and secure energy-efficient blockchain-based framework in IoT," *IEEE Access*, vol. 10, pp. 133682–133697, 2022.
[9] A. Fakieh and A. Akremi, "An effective blockchain-based defense model for organizations against vishing attacks," *Appl. Sci.*, vol. 12, no. 24, p. 13020, Dec. 2022.
[10] H. K. Alkahtani, K. Mahmood, M. Khalid, M. Othman, M. A. Duhayyim, A. E. Osman, A. A. Alneil, and A. S. Zamani, "Optimal graph convolutional neural network-based ransomware detection for cybersecurity in IoT environment," *Appl. Sci.*, vol. 13, no. 8, p. 5167, Apr. 2023.
[11] M. A. Alsoufi, S. Razak, M. M. Siraj, I. Nafea, F. A. Ghaleb, F. Saeed, and M. Nasser, "Anomaly-based intrusion detection systems in IoT using deep learning: A systematic literature review," *Appl. Sci.*, vol. 11, no. 18, p. 8383, Sep. 2021.

[12] N. A. Khan, G. Laouini, F. S. Alshammari, M. Khalid, and N. Aamir, "Supervised machine learning for jamming transition in traffic flow with fluctuations in acceleration and braking," *Comput. Electr. Eng.*, vol. 109, Jul. 2023, Art. no. 108740.
[13] M. Asif, S. Abbas, M. A. Khan, A. Fatima, M. A. Khan, and S.-W. Lee, "MapReduce based intelligent model for intrusion detection using machine learning technique," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 10, pp. 9723–9731, Nov. 2022.
[14] N. Alasmari, M. A. Alohali, M. Khalid, N. Almalki, A. Motwakel, M. I. Alsaid, A. E. Osman, and A. A. Alneil, "Improved metaheuristics with deep learning based object detector for intelligent control in autonomous vehicles," *Comput. Electr. Eng.*, vol. 108, May 2023, Art. no. 108718.
[15] 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.
[16] G. S. Aljumaie, G. H. Alzeer, R. K. Alghamdi, H. Alsuwat, E. Alsuwat, and N. Security, "Modern study on Internet of Medical Things (IOMT) security," *Int. J. Comput. Sci. Netw. Secur.*, vol. 21, no. 8, pp. 254–266, 2021.
[17] U. Khadam, M. M. Iqbal, M. Alruily, M. A. A. Ghamdi, and M. Ramzan, "Text data security and privacy in the Internet of Things: Threats, challenges, and future directions," *Wireless Commun. Mobile Comput.*, vol. 2020, Jan. 2020, Art. no. 7105625.
[18] S. Ramzan, A. Aqdus, V. Ravi, D. Koundal, R. Amin, and M. A. A. Ghamdi, "Healthcare applications using blockchain technology: Motivations and challenges," *IEEE Trans. Eng. Manag.*, vol. 70, no. 8, pp. 2874–2890, Aug. 2022.
[19] S. Shamshirband, M. Fathi, A. T. Chronopoulos, A. Montieri, F. Palumbo, and A. Pescapè, "Computational intelligence intrusion detection techniques in mobile cloud computing environments: Review, taxonomy, and open research issues," *J. Inf. Secur. Appl.*, vol. 55, Dec. 2020, Art. no. 102582.
[20] M. Alsafyani, F. Alhomayani, H. Alsuwat, and E. Alsuwat, "Face image encryption based on feature with optimization using secure crypto general adversarial neural network and optical chaotic map," *Sensors*, vol. 23, no. 3, p. 1415, Jan. 2023.
[21] A. H. Shahid and M. P. Singh, "Computational intelligence techniques for medical diagnosis and prognosis: Problems and current developments," *Biocybern. Biomed. Eng.*, vol. 39, no. 3, pp. 638–672, Jul. 2019.
[22] R. Amin, E. Rojas, A. Aqdus, S. Ramzan, D. Casillas-Perez, and J. M. Arco, "A survey on machine learning techniques for routing optimization in SDN," *IEEE Access*, vol. 9, pp. 104582–104611, 2021.
[23] M. A. Umer, K. N. Junejo, M. T. Jilani, and A. P. Mathur, "Machine learning for intrusion detection in industrial control systems: Applications, challenges, and recommendations," *Int. J. Crit. Infrastruct. Protection*, vol. 38, Sep. 2022, Art. no. 100516.
[24] A. Aqdus, R. Amin, S. Ramzan, S. S. Alshamrani, A. Alshehri, and E.-S. M. El-Kenawy, "Detection collision flows in SDN based 5G using machine learning algorithms," *Comput., Mater. Continua*, vol. 74, no. 1, pp. 1413–1435, 2023.
[25] H. Alqahtani, I. H. Sarker, A. Kalim, S. M. Hossain, and S. Ikhlaq, "Cyber intrusion detection using machine learning classification techniques," in *Proc. Int. Conf. Comput. Sci., Commun. Secur.*, Gujarat, India, 2020, pp. 121–131.
[26] F. Zhao, H. Zhang, J. Peng, X. Zhuang, and S.-G. Na, "A semi-self-taught network intrusion detection system," *Neural Comput. Appl.*, vol. 32, no. 23, pp. 17169–17179, Dec. 2020.
[27] P. Devan and N. Khare, "An efficient XGBoost–DNN-based classification model for network intrusion detection system," *Neural Comput. Appl.*, vol. 32, no. 16, pp. 12499–12514, Aug. 2020.
[28] X. Ling, S. Ji, J. Zou, J. Wang, C. Wu, B. Li, and T. Wang, "DEEPSEC: A uniform platform for security analysis of deep learning model," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 673–690.
[29] J. Newsome, B. Karp, and D. Song, "Polygraph: Automatically generating signatures for polymorphic worms," in *Proc. IEEE Symp. Secur. Privacy*, Oakland, CA, USA, Mar. 2005, pp. 226–241.
[30] N. Šrndic and P. Laskov, "Practical evasion of a learning-based classifier: A case study," in *Proc. IEEE Symp. Secur. Privacy*, Berkeley, CA, USA, May 2014, pp. 197–211.
[31] S. Choi, J. Lee, Y. Choi, J. Kim, and I. Kim, "Hierarchical network signature clustering and generation," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2016, pp. 1191–1193.
[32] 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.
[33] 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.
[34] S. Y. Mehr and B. Ramamurthy, "An SVM based DDoS attack detection method for ryu SDN controller," in *Proc. 15th Int. Conf. Emerg. Netw. Exp. Technol.*, New York, NY, USA, Dec. 2019, pp. 72–73.
[35] T. Su, H. Sun, J. Zhu, S. Wang, and Y. Li, "BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset," *IEEE Access*, vol. 8, pp. 29575–29585, 2020.
[36] K. Sethi, E. Sai Rupesh, R. Kumar, P. Bera, and Y. V. Madhav, "A context-aware robust intrusion detection system: A reinforcement learning-based approach," *Int. J. Inf. Secur.*, vol. 19, no. 6, pp. 657–678, Dec. 2020.
[37] Z. Li, V. Tam, and K. L. Yeung, "A coarse-to-fine grained knowledge refinement framework for network intrusion detection system," in *Proc. IEEE Region 10 Conf.*, Nov. 2022, pp. 1–6.
[38] A. Bagwari, M. Esmaeili, S. Goki, B. Masjidi, and M. Sameh, "ML-DDoSnet: IoT intrusion detection based on denial-of-service attacks using machine learning methods and NSL-KDD," *Wireless Commun. Mobile Comput.*, vol. 2022, Feb. 2022, Art. no. 84811452.

MOHAMMED A. AL GHAMDI received the bachelor's degree (Hons.) in computer science from King Abdul Aziz University, Jeddah, Saudi Arabia, in 2004, the master's degree (Hons.) in internet software systems from Birmingham University, Birmingham, U.K., in 2007, and the Ph.D. degree in computer science from the University of Warwick, U.K., in 2012. Since 2012, he has been with the Department of Computer Science, Umm Al-Qura University, Mecca, Saudi Arabia, as an Assistant Professor and then as an Associate Professor. He has authored more than 50 papers in international conferences and journals. He has published a number of good quality journal papers in machine learning, data analysis, AI, cloud computer, computer networks, and cybersecurity.