Optimized Blockchain-Based Deep Learning Model for Cloud Intrusion Detection

Sultan Alasmari
Department of Information System-College of Computer and Information Sciences,
Majmaah University, Majmaah 11952, Saudi Arabia
College of Technology and Business, Riyadh Elm University, King Fahad Road, Riyadh 12734, Saudi Arabia

***Abstract*—Cyberattacks are becoming increasingly complex and subtle. In many different types of networks, intrusion detection systems, or IDSs, are frequently employed to help in the prompt detection of intrusions. Blockchain technology has gained a lot of attention recently as a means of sharing data without a reliable third party. Specifically, it is impossible to change data stored in one block without changing all the following blocks. Create a deep learning (DL) method based on blockchain technology and hybrid optimization to improve the IDS's prediction accuracy. The UNSW-NB15 dataset is gathered via the Kaggle platform and utilized for Python system training. Principal component analysis (PCA) is used in the preprocessing to eliminate errors and duplication. Next, employ association rule learning (ARL) and information gain (IG) approaches to retrieve pertinent characteristics. The greatest features are the ones that improve detection performance through hybrid seahorse and bat optimization (HSHBA) selection. Lastly, create an efficient intrusion detection system by designing Blockchain-based Ensemble DL (BEDL) models, with convolutional neural networks (CNNs), restricted Boltzmann machines (RBM), and generative adversarial networks (GAN). The constructed model's experimental results are verified using pre-existing classifiers, yielding an improved accuracy of 99.12% and precision of 99%.**

***Keywords*—Intrusion detection system; blockchain; deep learning; hybrid optimization; cloud computing; feature selection**

I. Introduction

As the Internet has developed, an innovative technology known as the Internet of Things (IoT) has surfaced and been increasingly integrated into our daily lives. The IoT is directly empowering people and society through applications including healthcare, supply chain management, and RFID-based identity management systems [1]. By merging cloud computing and machine learning (ML), the base technology is starting to show promise for data analysis and modeling. Growth is being driven by the progress of IoT-based development across multiple industries. On the other hand, most IoT applications rely on centralized processing and storage architecture [2, 3]. There are number of security or privacy flaws in the centralized storage model. There are limitations in the underlying working paradigm that will make it difficult to expand IoT-based systems shortly. Decentralized storage models are required to solve these problems. Blockchain technology is one of the newest decentralized architectures [4, 5].

A collection of cryptographic entities can store and manage an assortment of time-stamped, unchanging data records in blocks using a technique called blockchain. The blocks are connected by the cryptographic hashes of the preceding block [6]. The timestamp, hash of the preceding block, and transaction comprise each block. Consequently, any modifications to the transactions need to be applied consistently through the consensus process across all of the blocks that comprise the blockchain [7]. This proves that the blockchain is the best data storage framework and ensures its immutability. Blockchain systems have been used for bitcoins, handling operations, financial services, healthcare, digital ID leadership, and many more uses. Two non-financial blockchain systems are Ethereum and Hyperledger [8, 9].

The proliferation of internet usage has facilitated data exchange and storage. Nevertheless, these data's susceptibility to hackers is greatly increased [10]. Numerous studies suggested the use of firewalls, data encryption, and user authentication to prevent unauthorized users from accessing stored data; yet, malevolent actors continue to find ways to circumvent these security measures and obtain illegal data access [11]. IDS has been proposed by further study to detect hostile intrusions in computer networks and internet-connected devices. Although intrusion detection systems have shown to be effective in detecting malicious activity, their limited perspective makes it difficult to spot coordinated or widespread cyberattacks [12, 13]. Because of this vantage point, certain attacks may go unnoticed or may not be discovered quickly enough. IDSs must swap attack features to quickly identify new assaults because certain attacks have managed to evade detection [14]. Thus, by combining the activities of participating IDS nodes, more harmful behaviors can be halted if IDS nodes communicate this threat information.

Protecting oneself from various kinds of attacks is more crucial than ever in the modern world, where data-centric research demands accurate DL algorithms [15]. Current studies have brought attention to ML systems' susceptibility to intrusion detection, whereby undetectable changes in the input data lead to inaccurate predictions in the output. A range of hostile attacks occur in real-time settings, and workable defense strategies are suggested to fend them off [16, 17]. In recent years, convolutional neural networks have been increasingly prevalent in the DL area as they have demonstrated remarkable performance in a wide range of application domains. High efficiency and long-term reliability are achieved using DL, which teaches neural network layers to see data as a hierarchical structure of principles [18]. When the amount of data increases, DL performs better than traditional ML. Blockchain-enabled DL-based intrusion detection algorithms have grown in

popularity and have been used for a wide range of applications in recent years [19, 20]. To improve IDS prediction and improve feature selection, the proposed IDS framework employed hybrid optimization and ensemble DL approaches. The key contribution of the developed model is followed as,

- UNSW-NB15 datasets are collected from the Kaggle website and trained in the system.
- Preprocessing steps include data cleaning, normalization, and dimensionality reduction to prepare the data for DL models.
- Then extract relevant features in the feature extraction phase.
- Moreover, select the most important features using the HSHBA model to enhance the detection results.
- Finally, intrusions are detected using the ensemble DL technique, and the output is predicted based on the majority voting of each classifier.
- To maintain the honesty of the blockchain, cryptographic hash functions can be employed.
- The experimental outcome is validated with existing techniques in terms of accuracy, precision, false rate, and so on to prove the efficiency.

Section II reviews the related study, Section III describes the problem definition, Section IV elaborates on the proposed model, and Section V depicts the experimentation analysis followed by the conclusion in Section VI.

II. Related Work

A deep blockchain framework (DBF) is proposed by Osama et al. [21] to provide privacy-based blockchain technology with intelligent agreements and security-based decentralized intrusion detection in Internet of Things networks. A DL algorithm called Bidirectional Long Short-Term Memory (BiLSTM) uses the IDS to handle sequential network information. Distributed intrusion detection systems are given privacy through the use of the Ethereum library in the development of privacy-based ledgers and smart contract techniques. Both users and cloud providers may find the framework useful as a decision-support tool.

A unique distributed intrusion detection system (IDS) that uses fog computing to identify DDoS attacks towards a mining facility in an IoT network supported by blockchain is proposed by Randhir et al. [22]. Random Forest (RF) and an improved gradient tree boosting system (XGBoost) are trained on dispersed fog nodes to assess performance. The BoT-IoT dataset, which comprises the majority of current attacks discovered in blockchain-enabled IoT networks, is used to evaluate the efficacy of the suggested methodology. In general, RF requires less time for testing and training on widespread fog nodes than XGBoost.

To improve IIoT security, Romany et al. [23] presented the efficient Blockchain-Assisted Cluster-based IDS method known as BAC-IDS. Clustering IoT devices is the goal of the BAC-IDS concept, which intends to enable blockchain-based safe data transfer and discover intrusions. The BAC-IDS method uses a clustering approach based on Harris Hawks Optimization (HHO) to select Cluster Heads (CH) and build clusters based on their preferences. The NSL-KDD2015 dataset shows that the BAC-IDS technique has a lower error rate (0.03) based on experimental results.

In a smart city, Internet of Things security is essential. IoT security is a serious concern because of the many objectives and significant drawbacks that impede the quick adoption of these smart gadgets. Using lightweight information technology, Erukala et al. [24] describe a permission-based blockchain network for safeguarding the key pairs of connected devices. Using the combined ML technique, a collaborative detection tool is utilized to identify DDoS attacks on Internet of Things devices. Next, include a blockchain system that securely distributes alarm warnings to every IoT network node with sufficiently secure identification.

To build, implement, and evaluate an intrusion detection system, Chao Liang et al. [25] presented a hybrid placement method based on a multi-agent framework, blockchain, and DL procedures. The modules that make up the system are reaction, analysis, data administration, and data collecting. The outcomes show how effective DL algorithms are in identifying transport layer threats. According to the experiment, DL algorithms can be used to detect intrusions in Internet of Things networks.

A blockchain-based DL and ML system was developed by Shraddha et al. [26] to enhance IDS performance. First, use a group of classifiers, like Random Forest, Convolutional Neural Network, and XGBoost, to identify anomaly assaults. Next, utilizing the blockchain system, identified assaults are transformed into signatures and transferred further throughout the network. In this step, information is stored and secured at a higher level using the cryptosystem that is a part of the blockchain. The suggested IDS system's experimental results show increased accuracy and greater detection performance.

To efficiently identify Advanced Persistent Threats (APT) attack characteristics on the terminals, Lampis [27] suggests an Intrusion Detection and Prevention System (BIDPS) supported by Blockchain technology. First, identify and stop the methods used by attackers, remove trust from the endpoint, and put it on-chain. To assess the BIDPS's efficacy, a testbed was constructed and more than 10 APT attack tactics were used to target the endpoint. Because the Blockchain is immutable, BIDPS can successfully fend off launched attacks, strengthening its detection and prevention operations.

To effectively detect attacks, Nour et al. [28] offer a federated DL-based intrusion detection system (FED-IDS), which transfers learning data from servers to dispersed vehicle edge nodes. To learn the spatial-temporal depictions of vehicle traffic flows required for categorizing various attack types, FED-IDS employs a context-aware transformer system. Miners validate distributed local upgrades on the blockchain to prevent erroneous updates from being added. The outcomes of the experiment demonstrate the viability of protecting intelligent transportation networking against cyberattacks.

III. Problem Statement

Considerable research has been done on the integration of intrusion detection and blockchain to enhance data privacy and identify current and potential threats, respectively. These methods use learning-based ensemble frameworks to protect data privacy while simultaneously making it easier to identify complicated harmful occurrences [29]. The most arduous and time-consuming task is preprocessing data. Thus, even a rookie researcher should be able to choose pertinent features from the dataset and apply them while creating an IDS model. A new technology called the Internet of Things (IoT) is being developed for the creation of numerous vital applications. These apps still use centralized storage design, though, so they face several serious issues, including single points of failure, privacy, and security. Critical IoT network failure points were made visible by a distributed denial of service (DDoS) attack on the cloud [30]. All of the devices have access to the Internet, making the collected heterogeneous IoT urban information more accessible to the public and more susceptible to intellectual property theft by hackers. Network data exchange requires the guarantee of security and secrecy. The majority of networks that hackers infiltrate are ones where data interchange is highly vulnerable.

A. Research Gap

In the realm of cloud intrusion detection, existing approaches primarily focus on either traditional ML methods or centralized DL models, both of which come with significant limitations. Traditional ML techniques struggle to handle the large volumes of diverse data generated in cloud environments and often fail to adapt to new or evolving attack patterns. Centralized DL models, while more capable of detecting complex intrusions, suffer from challenges related to scalability, single points of failure, and the potential for compromised data integrity. Additionally, the lack of a robust, tamper-resistant framework for sharing model updates or alerts across cloud nodes can undermine the reliability and trustworthiness of the system, especially in multi-tenant environments where data privacy and security are paramount. The proposed Optimized Blockchain-Based DL Model aims to fill these gaps by introducing a decentralized, blockchain-anchored architecture that leverages the power of DL for cloud intrusion detection while enhancing security, transparency, and scalability. Unlike centralized systems, this model allows distributed cloud nodes to collaboratively detect intrusions, with each node securely sharing model updates through smart contracts on the blockchain. This approach not only decentralizes the decision-making process but also ensures that model integrity is preserved across all participating nodes, eliminating the risk of single points of failure and enhancing the system's ability to handle real-time threats. Furthermore, by integrating blockchain, the proposed model introduces immutability and auditability in the detection process, addressing the trust and accountability gaps present in current systems. This work, therefore, marks a significant advancement in cloud security, providing a more resilient and transparent system for intrusion detection.

IV. Proposed Methodology

To stop network attacks, an IDS is essential in the field of cybersecurity. The choice engine being used determines how effective it is. A learning system built on blockchain and DL should be used to discover anomalies to boost the system's adaptability. This study offers an IDS based on DL and blockchain, utilizing optimization approaches to identify network threats. Protecting the accuracy and openness of the system depends on the decentralized and unchangeable ledger that blockchain technology offers. Everything about the intrusion detection process is recorded, including transactions and occurrences. Collaborative DL models, specifically CNNs, RBM, and GAN, can be utilized for intrusion detection assignments. The method used to develop these kinds of models are useful for detecting intrusions because they can identify intricate patterns and anomalies using network traffic data. Hash functions based on cryptography are also used in cloud environments to improve security and identify assaults.

A. Dataset Collection

The dataset used in this study to evaluate the suggested framework is the UNSW-NB15 [31]. There are 42 attributes in the UNSW-NB15, comprising three categorical inputs and 39 numerical inputs. The data formats (types) for the numerical inputs are binary, integer, and float. Two data subsets are also included in the UNSW-NB15: one for the training phase and another for testing. A thorough explanation of the attack's UNSW-NB15 subset distribution is given in Table I. The proposed technology is given in Fig. 1.

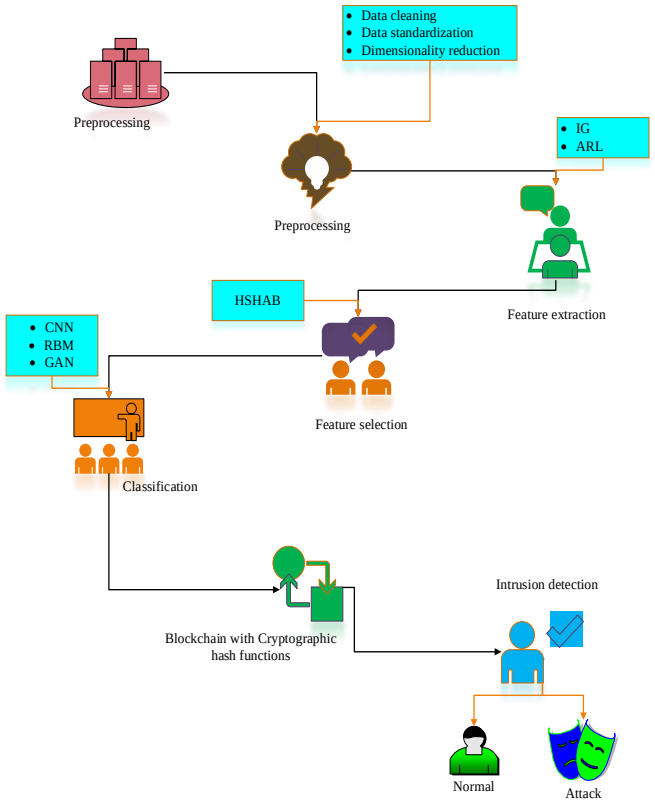


Fig. 1. Process of the developed technique.

TABLE I. DESCRIPTION OF THE UNSW-NB15 DATASET

Types of attack	Total No of data	No of training data	No of validation data	No of testing data	Data Attributes
Generic	40000	30081	9919	18871	1. Flow Features 2. Basic Features 3. Content Features 4. Time and Additional Features
Shellcode	1133	854	279	378	
Exploits	33393	25034	8359	11132	
Worms	130	99	31	44	
Reconnaissance	10491	7875	2616	3496	
Fuzzers	18184	13608	4576	6062	
Analysis	2000	1477	523	677	
Dos	12264	9237	3027	4089	
Backdoor	1746	1330	416	583	
Normal	56000	41911	14089	37000	

Divide the UNSW-NB15 training set into two parts: 25% of the initial training set is employed for validating the trained designs, and the remaining 75% of the original training is utilized to train the models. Furthermore, the validated models are tested using the UNSW-NB15-Test. The 10 assault classes that make up the UNSW-NB15. The data attributes if UNSW-NB15 include:

Flow Features: Attributes like dur (duration of the flow), proto (protocol used), and service (network service on the destination, e.g., HTTP, FTP, DNS).

Basic Features: These describe essential properties of the traffic, such as sbytes (source-to-destination bytes), dbytes (destination-to-source bytes), sttl (source time-to-live), and dttl (destination time-to-live).

Content Features: These focus on the content of packets, including features like sload (source packets per second) and dload (destination packets per second), which help in identifying abnormal patterns in data flows.

Time and Additional Features: These capture specific timing and behavior, such as ct_srv_src (connections to the same service from the same source) and ct_dst_ltm (connections to the same destination in the last 100 connections).

The dataset also includes a class label (label) that distinguishes between normal traffic (0) and attack traffic (1), with various attack types including DoS, backdoor, and shellcode. These attributes provide a detailed perspective of network traffic, enabling in-depth analysis for intrusion detection systems.

B. Preprocessing

One statistical method that makes use of an orthogonal change is PCA [32]. A set of correlated variables is transformed into a set of uncorrelated variables using PCA. For exploratory analysis of data, PCA is employed. Moreover, PCA can be used to investigate the connections between a set of variables. It can therefore be applied to reduce dimensionality.

Let's say a dataset $z^{(1)}, z^{(2)}, z^{(3)}, \ldots . z^{(n)}$ has n n-dimension data that must be transformed to dimension inputs i -dimension $(i << n)$ using PCA. Below is a description of PCA:

Data Standardization: Eq. (1) states that the raw data must have a unit variance & a zero mean.

$$k_j^i = \frac{k_j^i - \bar{k}_j}{\sigma_j} \forall j \qquad (1)$$

Using Eq. (2), determine the co-variance vector of the raw data.

$$\sum = \frac{1}{n}\sum_i^n \quad (k_i)(k_i)^t, \sum \in E^{n*n} \qquad (2)$$

Utilizing the formula from Eq. (3), determine the covariance matrix eigenvector and eigenvalue.

$$v^t \sum = \gamma\mu$$
$$V = [\ldots v_1\ v_{2\ldots.}\ v_n \ldots], v_i \in E^n \qquad (3)$$

It is necessary to project raw data onto a n -dimensional subspace: Top n The covariance matrix's eigenvector is selected. Eq. (4) provides the necessary vector calculation.

$$z_i^{new} = [v_1^t z^i\ v_2^t z^i \ldots\ldots\ldots\ldots v_k^t z^i\ ] \in E^k \qquad (4)$$

Thus, if the unprocessed data is with n dimensionality, It'll be condensed into new k data presented in three dimensions.

C. Feature Extraction

To quantify the valuable information extracted from the cleaned dataset, two techniques are employed: Information Gain (IG) and Associate Rule Learning (ARL). When it pertains to feature extraction, IG and ARL work well together to find pertinent traits that make a substantial contribution to the association rules.

1) Information Gain (IG): The value of a feature for forecasting the target variable is measured using IG [33]. When a particular characteristic is known, it computes the decrease in entropy or unpredictability of the target variable. IG is an entropy-based technique for evaluating features that quantify the amount of information a feature provides regarding the target class. IG can identify the features with the highest information based on the target class. Strongly related to the target class, features with high IG are typically chosen to get the greatest classification outcomes. Nevertheless, IG is unable to remove unnecessary features. The amount of information that was available before as well as after the attribute value was known usually determines how much information is gained. For multiple classes, IG can have a maximum value of 1. Eq. (5) provides the formula for entropy analysis of more than two classes.

$$G(Y) = \sum_{i=1}^k Q(y_i) Q\ (y_i) \qquad (5)$$

Let, Q is denoted as the number of classes. Moreover, feature $Y$of I_G and the class labels Z is designed in Eq. (6) and Eq. (7).

$$I_G(Y, Z) = G(Y) - G\left(\frac{Y}{Z}\right) \qquad (6)$$

$$G\left(\frac{Y}{Z}\right) = -\sum_j Q(z_i) \sum_i Q\left(\frac{y_i}{z_i}\right)\left(Q\left(\frac{y_i}{z_i}\right)\right) \quad (7)$$

Let, $G(Y)$is denoted as the entropy of Y and $G\left(\frac{Y}{Z}\right)$ is considered as entropy of Y after seeing Z. Since I_G is considered a filter technique, when dealing with big multidimensional data, it scales effectively.

2) Association rule learning: Using ARL [34], one can find intriguing correlations or interactions between variables in huge datasets. These fascinating relationships are typically concealed in unprocessed data, but if they are found and retrieved, they can be effectively used to describe the data. To aid in further identifying intrusions, associations among users and the applications they use must be extracted from the intriguing relationships and added to usage profiles that are both normal and suspicious.

The following is the official description of ARL: Let $K = \{k_1, k_2, \ldots, k_n\}$be a group of n features referred to as system audit data pieces. Let $D = \{d_1, d_2, \ldots . d_m\}$exist a group of entries in this dataset. Each record d_i has a subset of characteristics in K. An assumption of the kind in Eq. (8) is referred to as a rule.

$$R \Rightarrow S, \text{ where } S \subseteq K, \text{and } R \cap S = 0 \quad (8)$$

The subsequent item sets apply to intrusion detection when reading the aforementioned example by this definition: $R = d_1 = users$ and $S = d_2 = used\_users$. The suggestion $R \Rightarrow S$, is an ARL.

The subsequent item sets apply to intrusion detection when reading the aforementioned example by this definition: the support $Supp(R)$of an item set R, and the assurance $conf(R \Rightarrow S)$of a rule $R \Rightarrow S$as corresponds to Eq. (9).

$$conf(R \Rightarrow S) = \frac{Supp(R \cup S)}{Supp(R)} \quad (9)$$

Support for a set of objects $R\big(Supp(R)\big)$is the percentage of the data set's values that include the item set R. An algorithm for learning association rules consists of two distinct steps: First, determine an appropriate support level threshold and search the data for all likely common sets of items with support values higher than the threshold. Second, rules with confidence values higher than the minimal threshold are constructed using these acquired common item sets.

D. Feature Selection

Hybrid Sea horse and Bat Optimization are used to choose the dataset's best characteristics (HSHBA). It is the result of combining Bat Optimization (BO) [35] with Sea Horse Optimization (SHO) [36]. The movement, predatory behavior, and breeding of sea horses are simulated by the SHO algorithm. The core elements of SHO have three behaviors. The global and locaters techniques are adapted to the motion and hunting behaviors, respectively, to enhance the enhancements of the SHO algorithm. Here, Lévy flying is utilized to mimic the seahorse's movements as it spirals nearer to its greatest advantage. The mathematical relationship is described in Eq. (10).

$$Z^1_{new}(t+1) = Z_i(t) + Levy(\delta)\left(\left(Z_{elite}(t) - Z_i(t)\right) \times a \times b \times c + Z_{elite}(t)\right) \quad (10)$$

Let, $a, b,$ and c are denoted as coordinate vectors in three dimensions (a, b, c) in the spiraling motion, correspondingly. Eq. (11) is used to calculate the Brownian motion with waves.

$$\sigma = \left(\frac{\Gamma(1+\delta) \times sin sin\left(\frac{\pi\delta}{2}\right)}{\Gamma\left(\frac{1+\delta}{2}\right) \times \delta \times 2^{\left(\frac{\delta-1}{2}\right)}}\right) \quad (11)$$

Brownian motion is utilized to imitate the movement of the unit size of the sea horse, which is given by Eq. (12), to the left of the $\overrightarrow{r_1}$ cut-off point to enable to better comprehend the search space in SHO.

$$Z^1_{new}(t+1) = Z_i(t) + rand \times c \times \alpha_t \times (Z_i(t) - \alpha_t \times Z_{elite}) \quad (12)$$

Let, c is considered a constant coefficient, $\alpha_t$is represented as a random walk coefficient for Brownian motion.

Eq. (13) can be used to integrate these two conditions to determine the sea horse's new position at each repetition t.

$$Z^1_{new}(t+1) = \begin{cases} Z_i(t) + Levy(\delta)\left(\left(Z_{elite}(t) - Z_i(t)\right) \times a \times b \times c + Z_{elite}(t)\right) & \overrightarrow{r_1} > 0 \\ Z_i(t) + rand \times c \times \alpha_t \times (Z_i(t) - \alpha_t \times Z_{elite}) & \overrightarrow{r_1} \leq 0 \end{cases} \quad (13)$$

This stage involves updating each bat's position $\hat{x}_i$ and velocity $\hat{v}_i$ in a space of dimensions d. $\hat{x}_i$ and $\hat{v}_i$ ought to be updated afterward throughout the iterations. The new solutions $\hat{x}_i(t)$ and velocities $\hat{v}_i(t)$ at time step t. Eq. (14) shows the mathematical algorithm's hybrid efficiency.

$$Z^2_{new}(t+1) = \{\hat{x}_i(t-1) + \hat{v}_i(t)\ \overrightarrow{r_2} > 0.1\ (1-\eta) \times (Z^1_{new}(t) - rand \times Z_{elite}) + \eta \times Z^1_{new}(t)\ \overrightarrow{r_2} \leq 0.1\ \} \quad (14)$$

Let, $Z^1_{new}(t)$ shows the seahorse's updated location at that moment of t, $\overrightarrow{r_2}$ is denoted as a random number [0, 1], It is employed to modify the seahorse's duration of steps during predation, which gets shorter by a linear amount each iteration. Eq. (15) is used to determine the velocity.

$$\hat{v}_i(t) = \hat{v}_i(t-1) + (\hat{x}_i(t-1) - \hat{x}^*)\hat{f}_i \quad (15)$$

Let, β in the range of [0,1] is a vector chosen at random using a uniform range. Here, $\hat{x}^*$ is considered as the current best place (solution) in the world, which is found after evaluating every option among every i bat. Let, $\hat{f}_i$ is considered as frequency and is computed by applying Eq. (16).

$$\hat{f}_i = \hat{f}_{min} + (\hat{f}_{max} - \hat{f}_{min}).\ \beta \quad (16)$$

Let, $\hat{f}_{max}$ and $\hat{f}_{min}$ are denoted as maximum and minimum frequency. Instead, it indicates that the prey is moving more quickly than the seahorse was when it was hunting, allowing the prey to escape and the seahorse to fail to capture it. Using the HSHBA approach, the generated model chooses the best characteristics from the dataset. It improves the IDS system's performance in terms of detection and categorization. To detect intrusion, the chosen features are subsequently modified for the

classification stage. The HSHBA algorithm is described in Algorithm.1.

Algorithm:1 Feature select process using HSHSA
Start
{
Initialize Z_i
Compute fitness value// all seahorse
Determine Z_{elite} *//best seahorse*
While $(t < T)$*do*
If $(\overrightarrow{r_1} = random > 0)$***do*** *// movement*
{
Set constant parameter values
Execute Brownian motion using eqn. (11)
Sea horse position updated using eqn.10 and 12
}
Else if do
{
Update position using eqn. (13).
}
End if
Update bat position $\hat{x}_i$ *and velocity* $\hat{v}_i$ *// combine bat optimization*
{
Update the new position using eqn. (14) //new hybrid model
}
If ($\overrightarrow{r_2} > 0.1$***)***
{
Select best features
}
Else if ($\overrightarrow{r_2} \leq 0.1$***)***
{
Continue Searching
}
End if
Select best features
Enhance prediction accuracy
}
end

E. Classification

Ensemble DL models are employed at this stage to enhance the forecast outcomes. For intrusion detection tasks, the created EDL model incorporates convolutional neural networks (CNNs), Generative Adversarial Networks (GAN), and Restricted Boltzmann Machine (RBM). A detailed explanation is provided below.

1) CNN [37]: A multi-layer perceptron specifically created for identifying two-dimensional shapes is called a convolutional network. Convolution kernel characteristics and convolution layer connectivity weights are among the network parameters that are learned during the CNN training procedure. To determine the intrusions, the prediction procedure primarily uses the input data and network parameters. Eq. (17) and Eq. (18) are used to train the chosen features into the CNN algorithm's convolution layer.

$$x^{\leftrightarrow c}_j = f\left(v^{\leftrightarrow c}_j\right) \quad (17)$$

$$v^{\leftrightarrow c}_j = \sum_{i \in N_j} x^{\leftrightarrow}_i{}^{c-1} * h^{\leftrightarrow c}_{ij} + b^{\leftrightarrow c}_j \quad (18)$$

Let, $v^{\leftrightarrow c}_j$ is denoted as the net activation of the j^{th} network of the convolution layer c, it is obtained by removing the feature map produced by the preceding layer, and convolution averaging $x^{\leftrightarrow}_i{}^{c-1}$, $x^{\leftrightarrow}_i{}^{c}$ is denoted as the output of the j^{th} channel of the convolution layer c. $f(.)$ is considered an activation function and applies tanh and sigmoid operations, among others. N_j is a subset of the feature maps that are utilized as input to compute $v^{\leftrightarrow c}_j$, $h^{\leftrightarrow c}_{ij}$ is denoted as convolution kernel matrix, $b^{\leftrightarrow c}_j$ is denoted as convolution feature map with bias. Regarding a feature map output $x^{\leftrightarrow c}_j$, by every supplied feature map $x^{\leftrightarrow}_j{}^{c-1}$ might differ. $*$ is denoted as a convolution symbol. Then update $v^{\leftrightarrow c}_j$ into the pooling layer via Eq. (19).

$$v^{\leftrightarrow c}_j = \sum_{i \in N_j} \beta^{\leftrightarrow c}_j down\left(x^{\leftrightarrow}_j{}^{c-1}\right) + b^{\leftrightarrow c}_j \quad (19)$$

Let, $v^{\leftrightarrow c}_j$ is denoted as net activation of the j^{th} channel of the pooling layer c, It is produced by balancing and pooling the characteristics map output $x^{\leftrightarrow}_i{}^{c-1}$ of the previous layer, β is denoted as pooling layer weighting factor, $b^{\leftrightarrow c}_j$ is considered as pooling layer offset, $down(.)$ is denoted as pooling function. Eq. (20) and Eq. (21) are used to weigh the inputs and get the outcome via the activation function, which yields the output of the fully connected layer.

$$x^{\leftrightarrow c} = f(v^{\leftrightarrow c}) \quad (20)$$

$$v^{\leftrightarrow c} = w^{\leftrightarrow c} x^{\leftrightarrow c-1} + b^{\leftrightarrow c} \quad (21)$$

Let, $v^{\leftrightarrow c}$ is considered a fully connected layer net activation function c, it is acquired by removing and filtering the output map. $x^{\leftrightarrow c-1}$ is denoted as the previous layer. $w^{\leftrightarrow c}$ is considered as a fully connected network weight coefficient, and $b^{\leftrightarrow c}$ is considered a fully connected layer offset c.

2) RBM [38]: The RBM model has a visible layer $\tilde{v}$ with n units and a hidden layer $\tilde{h}$ with m units. In addition, a matrix of actual values $\tilde{w}_{n \times m}$ replicas the proportions of visible to hidden neurons, where $\tilde{w}_{ij}$ is denoted as the visible unit connection $\tilde{v}_i$ and the hidden unit $\tilde{h}_j$. The data is primarily received by the visible layer for processing, while its pattern and probability distribution are learned by the hidden layer. Furthermore, probably every layer's unit $\tilde{v}$ and $\tilde{h}$ are binary It came from the distribution of Bernoulli, i.e., $\tilde{v} \in \{0,1\}^n, \tilde{h} \in \{0,1\}^m$. Eq. (22) calculates an RBM's energy function:

$$E\left(\tilde{v}, \tilde{h}\right) = -\sum_{i=1}^{n} \tilde{x}_i \tilde{v}_i - \sum_{j=1}^{m} \tilde{y}_i \tilde{h}_j - \sum_{i=1}^{n} \sum_{j=1}^{m} \tilde{v}_i \tilde{h}_j \tilde{w}_{ij} \quad (22)$$

Let, x and y are denoted as hidden and visible unit biases. Furthermore, Eq. (23) simulates the combined likelihood of a specific configuration $\left(\tilde{v}, \tilde{h}\right)$:

$$P\left(\tilde{v}, \tilde{h}\right) = \frac{e^{-E(\tilde{v}, \tilde{h})}}{f_p} \quad (23)$$

Let, f_p is considered a partition function, which, while taking into account visible and hidden units, restores the chance over all conceivable configurations. Essentially, an RBM must become familiar with a set of parameters. using an algorithm for training. For every training set, it maximizes the sum of data possibilities ξ, which is described in Eq. (24)

$$argmax_{\Theta} \prod_{\tilde{v} \in \xi} P(\tilde{v}) \quad (24)$$

Implementing the negative of the logarithm functions, which is denoted by the negative log-likelihood (NLL), to describe this problem is an intriguing method. The NLL indicates the distribution estimation of the new information over the original data. Consequently, one can use the partial derivatives to calculate the derivatives of $\widetilde{W}$, $\tilde{x}$ and $\tilde{y}$at iteration t. The parameter updating rules are described by Eq. (25–27).

$$\widetilde{W}^{t+1} = \widetilde{W}^{t} + \eta \left(\tilde{v} P\left(\tilde{v}, \tilde{h}\right) - \tilde{v} P(v, h) \right) \quad (25)$$

$$\tilde{x}^{t+1} = \tilde{x}^{t} + (\tilde{v} - v) \quad (26)$$

$$\tilde{y}^{t+1} = \tilde{x}^{t} + \left(P\left(\tilde{v}, \tilde{h}\right) - P(v, h) \right) \quad (27)$$

Let, η is denoted as the learning rate, v is considered as the visible layer's reconstruction $\tilde{v}$, and h is considered as the hidden vector's estimation $\tilde{h}$ given v.

3) GAN [39]: A GAN consists of two parts, a generator g_e and a discriminator d_r, that compete with one another. g_e uses a noise vector as its input $\vec{n}$ and intends to provide high-quality fake data that closely approximates the original data. Moreover, d_r seeks to identify authentic data from artificially created data. The min-max goal function o_f is used to represent Eq. (28).

$$\min_{g_e} \max_{d_r} o_f\left(g_e, d_r\right) = E_{\vec{x}_r \sim P}\left[log\ log\ \left(d_r(\vec{x}_r) \right) \right] + E_{\vec{n} \sim M}\left[log\ log\ \left(1 - d_r\left(g_e(\vec{n}) \right) \right) \right] \quad (28)$$

Let, $\vec{x}_r \sim P$is considered as the actual distribution of data and $\vec{n} \sim M$ is denoted as Gaussian noise distribution. $d_r(\vec{x})$is denoted as outputs. The generator g_e gather $\vec{n}$ input to classify the intrusion.

a) Ensemble DL techniques: The maximum voting of each classifier determines which of the obtained prediction results is chosen. Max voting [40] entails gathering predictions for every class label and projecting which class label will receive the greatest number of votes using Eq. (29). Soft voting is an additional kind of maximum voting. In soft voting, Eq. (30) illustrates, that predicted chances are gathered for each class identity, and the class identity with the highest probability is predicted.

$$z' = [C_1'(\hat{x}), C_2'(\hat{x}), C_3'(\hat{x})] \quad (29)$$

Let, z' is denoted as the majority vote of each classifier, determines the class label C_1',C_2', and C_3'.

$$z' = argmax_i \sum_{j=1}^{n} W_j' P_{ij}' \quad (30)$$

Let, W_j' is considered as the weight that can be allocated to the j^{th}classifier.

F. Blockchain with Cryptographic Hash Function [41]

A blockchain's primary role is to offer a cryptographically safe method for collecting a permanent and globally verifiable collection of documents, known as blocks, that are systematically arranged by separate time stamps. Blockchains are commonly utilized as a distributed, open database of data transactions since they are frequently shared and synchronized via a peer-to-peer network. Every member of the blockchain network has access to the record data, which they can use to accept, reject, or verify using a consensus procedure. Records are added to the blockchain in the same sequence that they were verified after they are approved.

The foundation of blockchains is the cryptographically secure hash function, a fundamental building block of cryptography. These hashing algorithms $\widehat{H}: \{0,1\}^{*} \rightarrow \{0,1\}^{n}$map an input of any length to an output with a fixed length of n bits, and it needs to meet the security constraints listed below:

Preimage resistance: Considering a hash value $\hat{h}$, It ought to be necessary to $\Phi(2^n)$work involved with computing an $\hat{x}$ such that $\widehat{H}(\hat{x}) = \hat{h}$.

Second preimage resistance: the input $\hat{x}$ and hash value $\hat{h} = \widehat{H}(\hat{x})$ are needed in $O(2n)$ for computing $\hat{x}' \neq \hat{x}$ such that $H(x\ 0) = h.3)$.

Collision resistance: Need $\Phi(2^n)$determination to calculate any two $\hat{x}' \neq \hat{x}$ such that $\widehat{H}(\hat{x}') = \hat{h}$

The opponent does not influence the real hash value in collisions. (Second) preimage resistance is particularly important in the context of blockchains because attackers might change current blocks while maintaining the chain if they could identify second preimages with a specific mixfix. The aforementioned security criteria state that an attack of this kind needs to be at least 2^n for a hash function with n bits. The IDS determines each file's hash value when scanning records on a system and matches it to the store. If a similarity is discovered, malware is probably present.

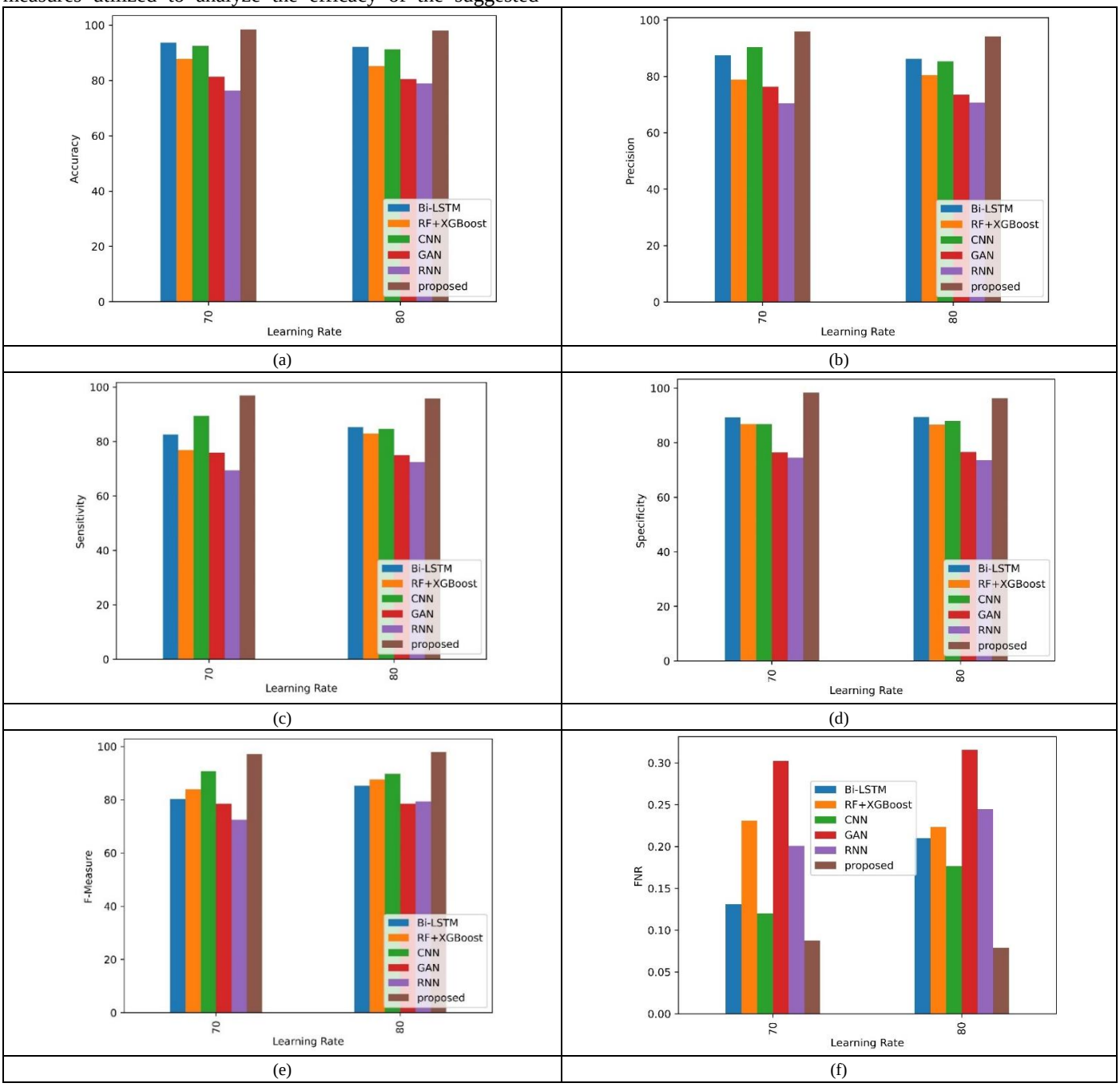
V. Results and Discussion

This study employed a variety of metrics to assess the efficacy of the proposed model. The BEDL model testing and training processes were conducted using Python. In the experiment, two learning rates 70 and 80 were chosen for the analysis of the study. The EDF method produced good results in many classification procedures. The presented method uses the BEDF model and hybrid optimization to increase the resilience of cloud computing. The architecture of proposed BEDL involves a decentralized network of cloud nodes that work collaboratively to detect intrusions. Each node in the blockchain network runs the DL IDS, where updates to the feature selection results are recorded immutably on the blockchain to ensure model integrity. The architecture comprises smart contracts that govern data sharing, model updates, and anomaly detection reporting across nodes. Smart Contracts are designed to trigger automatic actions such as initiating a model retraining process when new intrusion patterns are detected for accurate detections. Additionally, the smart contracts enforce data privacy by facilitating secure, encrypted communications between cloud nodes while ensuring that any detection-related alerts are stored immutably and transparently across the blockchain, guaranteeing auditability and trust. The use of consensus algorithms ensures that only validated model updates are propagated across the network, improving both security and collaboration in intrusion detection. Furthermore, the dataset used for investigation is

UNSW-NB15 which is available in https://www.kaggle.com/datasets/dhoogla/unswnb15. It is a benchmark dataset designed for evaluating IDS. It was created by simulating real-world network traffic at the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS), incorporating modern attack vectors. The dataset contains 49 features and 9 different attack categories, such as DoS, worms, backdoors, and exploits, alongside normal traffic. It provides both labeled and unlabeled data for training and testing machine learning models. The UNSW-NB15 dataset is widely used due to its diversity and realistic network behavior. The performance measures utilized to analyze the efficacy of the suggested technique are F1-score, False Positive Rate (FPR), False Negative Rate (FNR), Matthews Correlation Coefficient (MCC), Negative Predictive Value (NPV), accuracy, precision, sensitivity, and specificity.

A. Performance Analysis

Two learning rates 70 and 80 are employed for the training and testing of the developed technique. The experimental findings are tested against a variety of accessible DL classifiers, including Bi-LSTM [21], RF+XGBoost [23], RNN [40], CNN [37], and GAN [39].



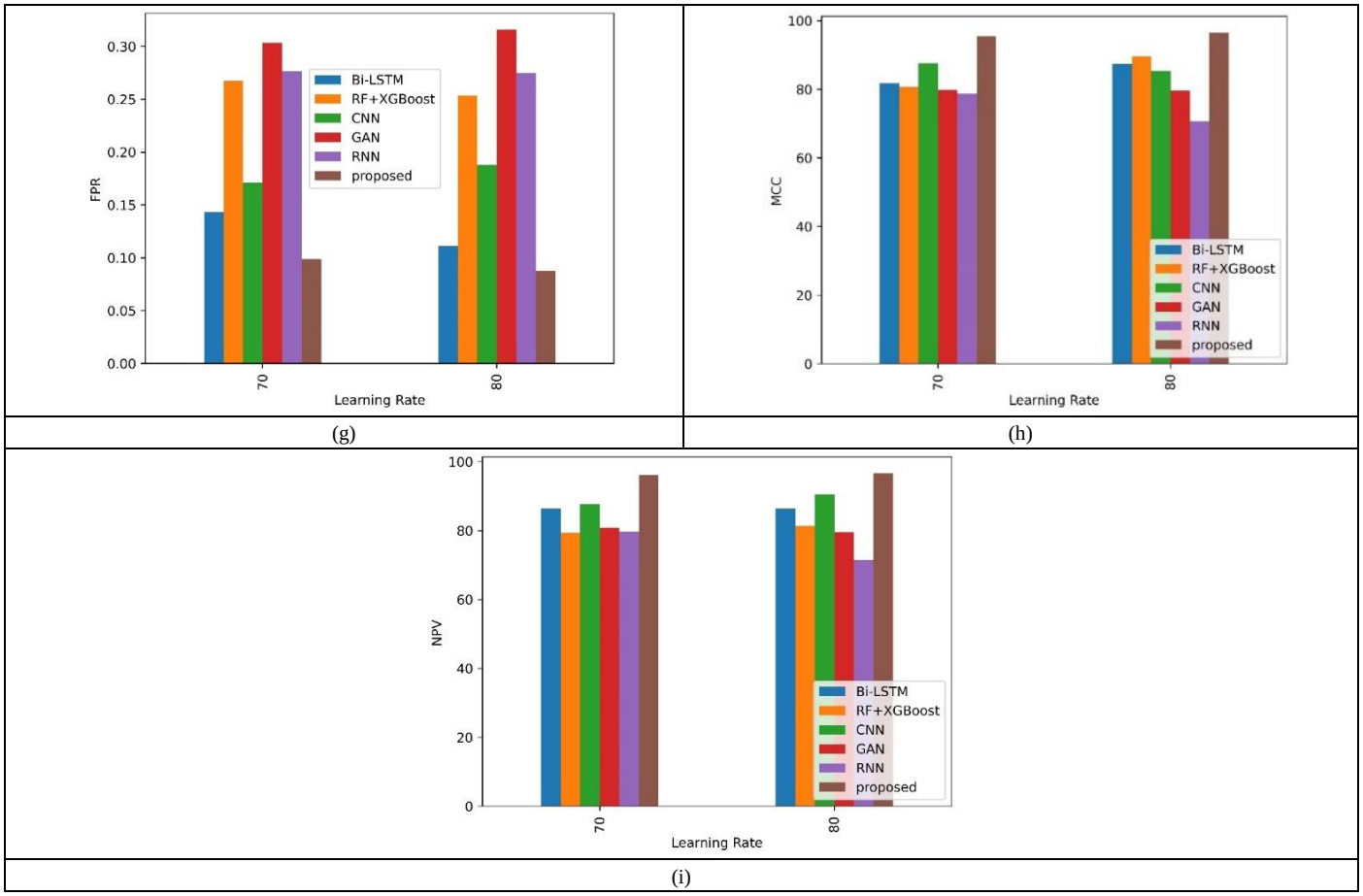


Fig. 2. Performance of proposed BEDL over Baseline Models for (a) Accuracy, (b) Precision, (c) Sensitivity. (d) Specificity, (e) F1-Score, (f) FNR, (g) FPR, (h) MCC, and (i) NPV.

1) Accuracy: The accuracy scores that several models obtained on the test are shown in Fig. 2(a), along with a suggested model, for two alternative scenarios: one with a learning rate of 70 and the other with a learning rate of 80. The Bi-LSTM model demonstrated a maximum accuracy of 93.655% at a learning rate of 70, followed by the CNN model's 92.4763% accuracy. The accuracy of the suggested model is 98.4763%. With an accuracy of 81.3652%, the GAN model was the least accurate of the models on the list; the RNN model was somewhat more accurate at 76.3764%. With an accuracy of 87.7766%, the ensemble methods RF+XGBoost fared better in the meantime. When comparing the accuracy ratings of all models in the first scenario to the second, which had a higher learning rate of 80, they generally declined. With 92.1121% accuracy, the Bi-LSTM model was still ahead of the CNN model, which came in second at 91.2344%. The suggested model saw a decline to 97.9987% but still displayed great accuracy. The accuracy of the ensemble methods RF+XGBoost similarly decreased, reaching 85.2235%. With the GAN model at 80.4762% and the RNN model at 78.8776%, the GAN and RNN models displayed similar patterns as in the prior situation.

2) Precision: The precision scores under two distinct learning rates (70 and 80) are shown in Fig. 2(b) for a variety of models, including Bi-LSTM, RF+XGBoost, CNN, GAN, RNN, and a proposed model. Precision is a metric that expresses the percentage of genuine positive predictions among all positive predictions, assessing how accurately a model predicts the future. The Bi-LSTM system achieves 87.54% precision at a learning rate of 70, while the RF+XGBoost model follows with 78.89% precision. 90.38% is the precision achieved by the CNN model, 76.37% by the GAN model, and 70.39% by the RNN model. Remarkably, with a precision of 95.97%, the suggested model beats all others, demonstrating its superiority in correctly predicting positive events at this learning rate. There is a tiny difference in the precision values below a learning rate of 80. The precision of the RF+XGBoost model rises to 80.48%, while that of the Bi-LSTM model falls to 86.23%. The precision of the GAN model falls to 73.48%, the RNN model stays almost the same at 70.77%, and the CNN model's precision reduces to 85.33%. In a similar vein, the precision of the suggested model drops but stays high at 94.22%.

3) Sensitivity: Sensitivity values for a range of models, including CNN, GAN, RNN, RF+XGBoost, Bi-LSTM, and a suggested model, are shown in Fig. 2(c) for various learning rates. True positive rate, another name for sensitivity, is the percentage of real positive cases that the model properly recognized. The suggested model beats the others with a

sensitivity of 96.88% at a learning rate of 70, demonstrating its higher capacity to accurately detect positive cases. The CNN and Bi-LSTM models, with corresponding sensitivity values of 89.48% and 82.66%, trail closely behind. With 76.88% and 69.47% sensitivity, respectively, RF+XGBoost and RNN outperform the GAN model, which comes in last with 75.99% sensitivity. The suggested model keeps its high sensitivity at 95.77% as the learning rate rises to 80, demonstrating its efficacy. Notably, the RF+XGBoost model outperforms the CNN and Bi-LSTM models in this configuration, with a sensitivity of 82.96%. However when compared to the other models, the GAN model's sensitivity is still the lowest at 74.99%, suggesting its relative weakness in correctly identifying positive cases.

4) Specificity: The specificity values of several models, as well as a suggested model under two distinct learning rates (70 and 80), are shown in Fig. 2(d). In binary classification, specificity is a metric that shows the percentage of real negative cases that the model correctly classifies as such. The Bi-LSTM model gets the maximum specificity of 89.37% under a learning rate of 70, followed by the CNN model at 87.99%. Additionally, the suggested model functions effectively, with a 98.35% specificity. It's important to keep in mind, though, that the GAN model performs comparatively worse than the others in terms of specificity, only reaching 76.48%. There are some variations in the models' performance when the learning rate is raised to 80. The Bi-LSTM model increases somewhat to 89.38% while maintaining its high specificity. There is also a minor improvement to 87.99% for the CNN model. Though it still performs noticeably better than most models, the suggested model's performance drops to 96.21%. Notably, as compared to its performance at the lower learning rate, the RNN model exhibits a drop in specificity.

5) F1-Score: The F-Measure performance scores under two distinct learning rates, 70 and 80, are displayed in Fig. 2(e) for a variety of models, including Bi-LSTM, RF+XGBoost, CNN, GAN, RNN, and a proposed model. The Bi-LSTM model obtains an F-Measure of 80.36% at a learning rate of 70, whereas RF+XGBoost does marginally better at 83.99%. CNN has the best performance, coming in at 90.78%, and the suggested model comes in at 97.23%. The F-Measure scores of GAN and RNN are lower, at 78.48% and 72.48%, respectively. A learning rate increase of 80 improves performance for the majority of models. Bi-LSTM sees a slight improvement to 85.23%, RF+XGBoost to 87.68%, CNN to 89.74%, and GAN and RNN to 79.39% and 78.56%, respectively. CNN experiences a slight decline. The suggested model demonstrates a significant rise to 97.99%.

6) FNR and FPR: The False Negative Rate (FNR) performance of the various models at the two learning rates 70 and 80 is displayed in Fig. 2(f). A distinct model, such as Bi-LSTM, RF+XGBoost, CNN, GAN, RNN, and a suggested model, is shown by each column. With a FNR of 0.087662, the suggested model beats all others at a learning rate of 70. CNN and Bi-LSTM both show comparatively low FNRs of 0.12004 and 0.13123, respectively. Nevertheless, models with higher FNR values, ranging from 0.20093 to 0.30234, include RF+XGBoost, GAN, and RNN. A discernible change in the models' performance occurs when the learning rate is raised to 80. The suggested model continues to have the lowest FNR (0.078876), indicating its resilience. Comparing the FNR values of Bi-LSTM, RF+XGBoost, and CNN to the 70-learning rate scenario, however, reveals a modest gain. Notably, with values over 0.24, GAN and RNN continue to show greater FNRs than the other models. On the other hand, all models perform somewhat better when the learning rate is raised to 80. Notably, the suggested model keeps the lowest FPR, demonstrating its superiority over the other models even more. Additionally, Bi-LSTM exhibits a significant drop in FPR in Fig. 2(g), demonstrating its sensitivity to variations in learning rate. The FPRs of RF+XGBoost, CNN, and GAN are comparable, while the performance of RNN is largely constant.

7) MCC: The scores for several models and a suggested model are shown in Fig. 2(h) for two separate scenarios: one with a learning rate of 70 and the other with a learning rate of 80. The models perform differently in the first situation when the learning rate is 70. The suggested model achieves 95.56553, while the Bi-LSTM model reaches 81.8009, RF+XGBoost at 80.6775, CNN at 87.6544, GAN at 79.7668, and RNN at 78.74662. Interestingly, the suggested model performs noticeably better than the others, demonstrating its usefulness in this situation. Significant differences in the model's performance can be seen in the second scenario, which uses a higher learning rate of 80. The Bi-LSTM model outperforms RF+XGBoost at 89.657, with an MCC of 87.3766. Nevertheless, the performance of the GAN and RNN models further declines to 79.65564 and 70.6553, respectively, while the CNN model's performance reduces to 85.4773. With an MCC of 96.4878, the suggested model maintains its strong performance despite these modifications, demonstrating its resilience and superiority over the other models in this situation.

8) NPV: The performance measures, namely the net present value (NPV), of various models in two distinct situations are displayed in Fig. 2(i) one with a learning rate of 70 and the other with an 80. A suggested model, CNN, GAN, RNN, RF+XGBoost, Bi-LSTM, and RNN are among the models that are compared. The suggested model performs better than other models in the first scenario with a 70-learning rate, obtaining a value of 6.1232. With an NPV of 87.6598, CNN outperforms the others, while Bi-LSTM, with an NPV of 86.3624, is not far behind. The NPV values of RF+XGBoost, GAN, and RNN are 79.3765, 80.87763, and 79.68773. With an NPV of 96.5886, the suggested model maintains its advantage in the second scenario with an 80-learning rate. CNN outperforms the first scenario by a substantial margin, obtaining the highest NPV among the models after the suggested one, 90.4874. Next, with an NPV of 81.3885, is RF+XGBoost. But with an NPV of 71.3874, RNN's performance significantly deteriorates, and it becomes the least-performing model out of all of them. To enhance the robustness and security of the cloud

environment, a cryptographic hash function is generated using blockchain. The gained experimental outcomes are compared with existing classifiers and attained better experimental outcomes. The designed technique gained accuracy of 98.47%, and 97.99% for 70 and 80 learning rates, and also gained less FPR of 0.098, and 0.087 for 70 and 80 learning rates. The developed technique improves the performance and IDS and enhances the efficiency and robustness by using blockchain. In the future, improving blockchain networks' scalability is essential to enabling the widespread deployment of IDS. Subsequent investigations may concentrate on creating innovative consensus processes or layer-2 scaling approaches to manage the growing number of events and transactions produced by IDS sensors.

B. Real-time Implementation Model

A pilot deployment is conducted across a distributed cloud infrastructure. The system's performance is assessed by simulating various intrusion scenarios, including DDoS attacks [41], unauthorized access attempts, and insider threats. Key performance indicators such as detection accuracy, false positive/negative rates, latency in intrusion detection, and blockchain transaction throughput are monitored. Additionally, the scalability of the system is evaluated by increasing the number of cloud nodes and analyzing the consensus efficiency, smart contract execution times, and resource utilization. The evaluation also considers the impact of network delays, data privacy enforcement, and system robustness in handling high-traffic environments, ensuring the solution's practicality and effectiveness for real-world cloud security.

When a DDoS attack occurs targeting the platform hosted on cloud system, the proposed BEDL model detects abnormal traffic spikes. Once an anomaly is detected, a smart contract is triggered, validating the threat and broadcasting it across the blockchain. This ensures all nodes in the network are aware of the attack, preventing it from spreading further. The smart contract also records the event immutably for future audits and triggers automated actions such as load balancing and firewall rule updates to mitigate the threat in real-time, enhancing both the security and resilience of the cloud infrastructure.

VI. Conclusion

This paper designs blockchain-based DL models to enhance the security of cloud computing. Three types of DL techniques are combined such as CNN, GAN, and RBM to enhance the prediction results of the developed model. UNSW-NB15 dataset is collected and they are cleaned, standardized, and reduced dimensionality using preprocessing. They select the best features to improve the attack prediction rate using the HSHBA model. Additionally, detect the intrusion present in the cloud using a blockchain-based EDL model. The final results are predicted based on the majority and soft voting of the designed technique. Future research could explore integrating federated learning [42] with blockchain to further enhance data privacy in cloud intrusion detection, allowing decentralized model training without sharing sensitive data. Additionally, adopting quantum-safe cryptography in the blockchain layer could future-proof the system against quantum computing threats. The scalability of the proposed architecture can be improved through layer-2 blockchain solutions to reduce latency. Research can also focus on adaptive DL models that evolve with emerging threats in real time. Lastly, expanding the system's application to edge computing environments could enhance security in IoT-based cloud ecosystems.

Acknowledgment

The author extends the appreciation to the Deanship of Postgraduate Studies and Scientific Research at Majmaah University for funding this research work through the project number (R-2024-1245).

References

[1] V. Saravanan, M. Madiajagan, S.M. Rafee, P. Sanju, T.B. Rehman, and B. Pattanaik, "IoT-based blockchain intrusion detection using optimized recurrent neural network," Multimedia Tools and Applications, pp. 1-22, 2023.

[2] A.A. Khan, M.M. Khan, K.M. Khan, J. Arshad, and F. Ahmad, "A blockchain-based decentralized machine learning framework for collaborative intrusion detection within UAVs," Computer Networks, vol. 196, p. 108217, 2021.

[3] R. Kumar, P. Kumar, R. Tripathi, G.P. Gupta, N. Kumar, and M.M. Hassan, "A privacy-preserving-based secure framework using blockchain-enabled deep-learning in cooperative intelligent transport system," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 9, pp. 16492-16503, 2021.

[4] S. Badri, "HO-CER: Hybrid-optimization-based convolutional ensemble random forest for data security in healthcare applications using blockchain technology," Electronic Research Archive, vol. 31, no. 9, pp. 5466-5484, 2023.

[5] S. Thiruvenkatasamy, R. Sivaraj, and M. Vijayakumar, "Blockchain Assisted Fireworks Optimization with Machine Learning based Intrusion Detection System (IDS)," Tehnički vjesnik, vol. 31, no. 2, pp. 596-603, 2024.

[6] I. Katib and M. Ragab, "Blockchain-assisted hybrid harris hawks optimization based deep DDoS attack detection in the IoT environment," Mathematics, vol. 11, no. 8, p. 1887, 2023.

[7] D.K. Jain, W. Ding, and K. Kotecha, "Training fuzzy deep neural network with honey badger algorithm for intrusion detection in cloud environment," International Journal of Machine Learning and Cybernetics, vol. 14, no. 6, pp. 2221-2237, 2023.

[8] L. Almuqren, K. Mahmood, S.S. Aljameel, A.S. Salama, G.P. Mohammed, and A.A. Alneil, "Blockchain Assisted Secure Smart Home Network using Gradient Based Optimizer with Hybrid Deep Learning Model," IEEE Access, 2023.

[9] O. Shende, R.K. Pateriya, P. Verma, and A. Jain, "CEBM: collaborative ensemble blockchain model for intrusion detection in IoT environment," 2021.

[10] E. Ntizikira, L. Wang, J. Chen, and K. Saleem, "Honey-block: Edge assisted ensemble learning model for intrusion detection and prevention using defense mechanism in IoT," Computer Communications, vol. 214, pp. 1-17, 2024.

[11] S.K. Kanna, M.Y.B. Murthy, M.B. Gawali, S.M. Rubai, N.S. Reddy, G. Brammya, and N.S. Preetha, "A deep learning-based disease diagnosis with intrusion detection for a secured healthcare system," Knowledge and Information Systems, pp. 1-39, 2024.

[12] S.K. Gupta, M. Tripathi, and J. Grover, "Hybrid optimization and deep learning based intrusion detection system," Computers and Electrical Engineering, vol. 100, p. 107876, 2022.

[13] A. Aljabri, F. Jemili, and O. Korbaa, "Convolutional neural network for intrusion detection using blockchain technology," International Journal of Computers and Applications, vol. 46, no. 2, pp. 67-77, 2024.

[14] W. Dhifallah, T. Moulahi, M. Tarhouni, and S. Zidi, "Intellig_block: Enhancing IoT security with blockchain-based adversarial machine learning protection," International Journal of Advanced Technology and Engineering Exploration, vol. 10, no. 106, p. 1167, 2023.

[15] S. Siddamsetti and M. Srivenkatesh, “Implementation of Blockchain with Machine Learning Intrusion Detection System for Defending IoT Botnet and Cloud Networks,” Ingénierie des Systèmes d'Information, vol. 27, no. 6, 2022.

[16] A.A. Sharadqh, H.A.M. Hatamleh, S.S. Saloum, and T.A. Alawneh, “Hybrid chain: blockchain enabled framework for bi-level intrusion detection and graph-based mitigation for security provisioning in edge assisted IoT environment,” IEEE Access, vol. 11, pp. 27433-27449, 2023.

[17] E. Ashraf, N.F. Areed, H. Salem, E.H. Abdelhay, and A. Farouk, “FIDChain: Federated intrusion detection system for blockchain-enabled IoT healthcare applications,” In Healthcare, vol. 10, no. 6, p. 1110, June 2022.

[18] R. Kumar, P. Kumar, M. Aloqaily, and A. Aljuhani, “Deep-learning-based blockchain for secure zero touch networks,” IEEE Communications Magazine, vol. 61, no. 2, pp. 96-102, 2022.

[19] A. Albakri, B. Alabdullah, and F. Alhayan, “Blockchain-assisted machine learning with hybrid metaheuristics-empowered cyber attack detection and classification model,” Sustainability, vol. 15, no. 18, p. 13887, 2023.

[20] E.M. Onyema, S. Dalal, C.A.T. Romero, B. Seth, P. Young, and M.A. Wajid, “Design of intrusion detection system based on cyborg intelligence for security of cloud network traffic of smart cities,” Journal of Cloud Computing, vol. 11, no. 1, p. 26, 2022.

[21] O. Alkadi, N. Moustafa, B. Turnbull, and K.K.R. Choo, “A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks,” IEEE Internet of Things Journal, vol. 8, no. 12, pp. 9463-9472, 2020.

[22] R. Kumar, P. Kumar, R. Tripathi, G.P. Gupta, S. Garg, and M.M. Hassan, “A distributed intrusion detection system to detect DDoS attacks in blockchain-enabled IoT network,” Journal of Parallel and Distributed Computing, vol. 164, pp. 55-68, 2022.

[23] R.F. Mansour, “Blockchain assisted clustering with intrusion detection system for industrial internet of things environment,” Expert Systems with Applications, vol. 207, p. 117995, 2022.

[24] E.S. Babu, B.K.N. SrinivasaRao, S.R. Nayak, A. Verma, F. Alqahtani, A. Tolba, and A. Mukherjee, “Blockchain-based Intrusion Detection System of IoT urban data with device authentication against DDoS attacks,” Computers and Electrical Engineering, vol. 103, p. 108287, 2022.

[25] C. Liang, B. Shanmugam, S. Azam, A. Karim, A. Islam, M. Zamani, S. Kavianpour, and N.B. Idris, “Intrusion detection system for the internet of things based on blockchain and multi-agent systems,” Electronics, vol. 9, no. 7, p. 1120, 2020.

[26] S.R. Khonde and V. Ulagamuthalvi, “Blockchain: Secured Solution for Signature Transfer in Distributed Intrusion Detection System,” Computer Systems Science & Engineering, vol. 40, no. 1, 2022.

[27] L. Alevizos, M.H. Eiza, V.T. Ta, Q. Shi, and J. Read, “Blockchain-enabled intrusion detection and prevention system of APTs within zero trust architecture,” IEEE Access, vol. 10, pp. 89270-89288, 2022.

[28] M. Abdel-Basset, N. Moustafa, H. Hawash, I. Razzak, K.M. Sallam, and O.M. Elkomy, “Federated intrusion detection in blockchain-based smart transportation systems,” IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 3, pp. 2523-2537, 2021.

[29] G.G. Gebremariam, J. Panda, and S. Indu, “Blockchain-based secure localization against malicious nodes in IoT-based wireless sensor networks using federated learning,” Wireless communications and mobile computing, 2023.

[30] R. Kumar, P. Kumar, R. Tripathi, G.P. Gupta, A.N. Islam, and M. Shorfuzzaman, “Permissioned blockchain and deep learning for secure and efficient data sharing in industrial healthcare systems,” IEEE Transactions on Industrial Informatics, vol. 18, no. 11, pp. 8065-8073, 2022.

[31] https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15

[32] G.T. Reddy, M.P.K. Reddy, K. Lakshmanna, R. Kaluri, D.S. Rajput, G. Srivastava, and T. Baker, “Analysis of dimensionality reduction techniques on big data,” IEEE Access, vol. 8, pp. 54776-54788, 2020.

[33] D. Stiawan, M.Y.B. Idris, A.M. Bamhdi, and R. Budiarto, “CICIDS-2017 dataset feature analysis with information gain for anomaly detection,” IEEE Access, vol. 8, pp. 132911-132921, 2020.

[34] S. Sivanantham, V. Mohanraj, Y. Suresh, and J. Senthilkumar, “Association Rule Mining Frequent-Pattern-Based Intrusion Detection in Network,” Computer Systems Science & Engineering, vol. 44, no. 2, 2023.

[35] X.S. Yang and A. Hossein Gandomi, “Bat algorithm: a novel approach for global engineering optimization,” Engineering computations, vol. 29, no. 5, pp. 464-483, 2012.

[36] F.A. Özbay, “A modified seahorse optimization algorithm based on chaotic maps for solving global optimization and engineering problems,” Engineering Science and Technology, an International Journal, vol. 41, p. 101408, 2023.

[37] J. Cheng, Y. Liu, X. Tang, V.S. Sheng, M. Li, and J. Li, “DDoS Attack Detection via Multi-Scale Convolutional Neural Network,” Computers, Materials & Continua, vol. 62, no. 3, 2020.

[38] G.H.D. Rosa, M. Roder, D.F. Santos, and K.A. Costa, “Enhancing anomaly detection through restricted Boltzmann machine features projection,” International Journal of Information Technology, vol. 13, pp. 49-57, 2021.

[39] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F.Y. Wang, “Generative adversarial networks: introduction and outlook,” IEEE/CAA Journal of Automatica Sinica, vol. 4, no. 4, pp. 588-598, 2017.

[40] M.A. Khan, “HCRNNIDS: Hybrid convolutional recurrent neural network-based network intrusion detection system,” Processes, vol. 9, no. 5, p. 834, 2021.

[41] A. I.Gide, and A.A. Mu’azu, "A Real-Time Intrusion Detection System for DoS/DDoS Attack Classification in IoT Networks Using KNN-Neural Network Hybrid Technique", Babylonian Journal of Internet of Things, pp. 60-69, 2024.

[42] R. T. Hameed, and O. A. Mohamad, "Federated Learning in IoT: A Survey on Distributed Decision Making", Babylonian Journal of Internet of Things, Vol. 2023, pp. 1-7, 2023.