mathematics

MDPI

Article

A Double-Layer Indemnity Enhancement Using LSTM and HASH Function Technique for Intrusion Detection System

Abdullah Marish Ali [1], Fahad Alqurashi [1], Fawaz Jaber Alsolami [1] and Sana Qaiyum [2,*]

[1] Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia
[2] School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA 70504, USA
* Correspondence: sqaiyum.cs@gmail.com

Abstract: The Intrusion Detection System (IDS) is the most widely used network security mechanism for distinguishing between normal and malicious traffic network activities. It aids network security in that it may identify unforeseen hazards in network traffic. Several techniques have been put forth by different researchers for network intrusion detection. However, because network attacks have increased dramatically, making it difficult to execute precise detection rates quickly, the demand for effectively recognizing network incursion is growing. This research proposed an improved solution that uses Long Short-Term Memory (LSTM) and hash functions to construct a revolutionary double-layer security solution for IoT Network Intrusion Detection. The presented framework utilizes standard and well-known real-time IDS datasets such as KDDCUP99 and UNSWNB-15. In the presented framework, the dataset was pre-processed, and it employed the Shuffle Shepherd Optimization (SSO) algorithm for tracking the most informative attributes from the filtered database. Further, the designed model used the LSTM algorithm for classifying the normal and malicious network traffic precisely. Finally, a secure hash function SHA3-256 was utilized for countering the attacks. The intensive experimental assessment of the presented approach with the conventional algorithms emphasized the efficiency of the proposed framework in terms of accuracy, precision, recall, etc. The analysis showed that the presented model attained attack prediction accuracy of 99.92% and 99.91% for KDDCUP99 and UNSWNB-15, respectively.

Keywords: network system; Internet of Things; malicious event identification; artificial intelligence; optimization; hash function

MSC: 68T05

Citation: Ali, A.M.; Alqurashi, F.; Alsolami, F.J.; Qaiyum, S. A Double-Layer Indemnity Enhancement Using LSTM and HASH Function Technique for Intrusion Detection System. *Mathematics* **2023**, *11*, 3894. https://doi.org/10.3390/math11183894

Academic Editor: Hongyu LIU

Received: 2 August 2023
Revised: 29 August 2023
Accepted: 5 September 2023
Published: 13 September 2023

1. Introduction

Advancements in information technology have transformed ways of data gathering and exchange. This technology has led to unprecedented knowledge sharing and connectivity, enabling individuals and organizations to tap into a wealth of information instantly. In addition, it helps to manage personal networks, online platforms, and other social media networks, enabling users to connect, learn, and collaborate globally. However, these advancements have introduced certain challenges and issues, including spam, viruses, malicious activity, etc. Therefore, designing a solution is important for mitigating these issues before they interrupt the network performance. Currently, healthcare units are adopting these advanced technologies to ensure patient care and instant treatment planning. Conventional medicine, which has biotechnology at its foundation, has progressively started to digitalize and inform itself, in line with a scientific hypothesis and technological innovation [1]. Currently, this advanced information technology is utilized in various domains, such as smart healthcare, smart home, industrial IoT, etc. [2]. By adopting major enabling technologies, including the IoT, blockchain, artificial intelligence

(AI), and next-generation wireless networks (5G and 6G), the smart healthcare industry has undergone a significant transition [3,4]. These innovations have been used by the healthcare sector, which has enhanced patient care and quality of life [5]. The interaction between patients and healthcare specialists is improved by applying IoT technologies in the healthcare industry [6]. It allows portable devices (sensor nodes) to swiftly communicate patient biometric data to other sensor nodes, or remotely to healthcare professionals such as doctors, pharmacists, and labs. This data can include such information as temperature, oxygen saturation, blood pressure, and more [7].

The sensors include a CPU, memory, feature profiles, and data profiles. The feature profile specifies the designer, kind, measurement region, manufacturing date, and sensor position [8]. The data profile handles the data format that the sensors are producing. If negotiations occur to ensure that the stated communication can be handled without intrusions or assaults, the healthcare organization will be insecure [9]. The most important technology for network security is intrusion detection, which allows for interactive and real-time network surveillance [10]. An IDS was developed to discover problems, vulnerabilities, and accomplishments on a target host. Also, an IDS's primary objective is to detect hazards so that they may be put out of the spectrum by a network's design and prevent interference with data being sent and received [11].

Despite the tremendous research programs, IDS still needs help with detection performance while reducing warning rates and recognizing disruption [12]. An efficient way to combat the rising tide of cyberattacks plaguing the healthcare ecosystem is to implement AI expertise in smart healthcare systems [13]. Several researchers have suggested using AI in the smart healthcare system as a reliable and practical security solution [14]. Many of the relevant studies in the literature have developed computerized simulations for network intrusion recognition using machine learning techniques, such as K-nearest neighbor (KNN) [15], Naïve Bayes [16], Support Vector Machine (SVM) [17], Random Forest (RF) [18], etc.

Moreover, Deep Neural Networks (DNN) [19], Convolutional Neural Networks (CNN) [20], Long Short-Term Memory (LSTM) [21], and others are examples of widely used deep learning techniques. For small amounts of data, machine learning algorithms work well enough. However, the network generates a large quantity of data in real time, and current machine-learning algorithms are unable to learn such a vast amount of data [22]. Additionally, data is subjected to minimal training and utilized as a benchmark. Deep learning algorithms have a data-hungry nature. Recently, deep learning (DL) structures are widely applied in IDS to identify attacks throughout the network [23] efficiently. Since deep learning approaches can adapt to enormous amounts of data, they are used more often [24]. Additionally, it uses a complex architecture that introduces nonlinear variations to get considerable level reflections in the information, giving it a high identification rate. Because deep learning can adapt to massive amounts of data, it is used increasingly often [25]. Moreover, the attack prevention algorithm is necessary after IDS. In addition to being efficient when applied in hardware and programming using Boolean operations, it is also inexpensive when compared with various security techniques [26]. For safeguarding embedded components, such as sensors, in Internet of Things (IoT) applications that employ symmetric key-based message authentication codes (MACs), the SHA3-256-single way function is excellent. The algorithm performs better than its forerunners, with an Intel Core 2 processor reporting an average speed of 12.5 cycles per byte. As a result, the current study's objective is to investigate numerous security assaults and their defenses for the smart healthcare environment while also supplying the security algorithm. The major contribution of this research is provided as follows:

(a) Data collecting is offered through the IoT network. The collected data should be normalized at the pre-processing stage.
(b) To shorten training time and improve classification performance, it is necessary to provide an effective and ideal feature selection process for IDS using SSO.
(c) Moreover, the deep LSTM method is used to classify packets (regular, malicious traffic).

(d) The system classifies different kinds of attack packets as Denial of Service (DoS), User to Root (U2R), Probe, R2L, and Anomaly.
(e) Also, a SHA3-256-based single-way hash function with Homomorphic Encryption (HE) method is designed for IoT intrusion prevention.
(f) Assess the solution using suitable result parameters to determine the overall efficacy of the developed approach.

The residue of the work is systematized as follows: In Section 2, we deliberated the IDS-associated work and some of the public datasets. The dataset, the model, and the procedure are described in Section 3. Section 4 provides the results. Section 5 presents the conclusion and forthcoming work.

2. Related Work

The following are a few articles interconnected to the presented study. To detect intrusions in the IoT-Cloud context, Retnaswamy and Bharathi [27] proposed using a Morlet Wavelet Kernel Function with an LSTM (MWKF-LSTM)-based classifier. A SHA-512 hashing method integrating a blockchain authorization (SHABA) framework is created and verifies each device or user's legitimacy before allowing information to updated to the cloud. The best Feature Selection (FS) model is the Differential Evaluation-assisted Dragonfly Algorithm (DEDFA), which improves classification performance. The non-injected data was encoded and stored securely in the cloud, employing the Enhanced Elliptical Curve Cryptography (E2CC) technique after ID. The user's authentication is then once again verified during the data retrieval stage to protect user privacy, which keeps outsiders from accessing the cloud's encrypted data. The key findings of this method are accuracy, FPR, encryption, and decryption time. More security assaults can be analyzed further.

Integrating a DL process, an IDS methodology was created using bi-directional LSTM (Bi-LSTM) as the main emphasis of Pooja, T. S., and Purohit Shrinivasacharya's [28] research. In tests, the developed system is verified using the KDDCUP99 and UNSW-NB15 databases. With 99% accuracy for both databases, the perfect exploiting Bi-LSTM fashioned excellent outcomes. Bi-directional LSTM outperforms other methods, according to the data.

M. Islabudeen and M. K. Kavitha Devi [29] projected a Smart Algorithm for ID and Prevention System (SA-IDPS) to diminish outbreaks in MANETs by machine learning techniques. By utilizing the One-Way Hash Chain Operation, movable operators are initially registered with the Reliable Expert. This was accomplished using a type 2 fuzzy controller that deliberates packet heading material. Mutual data is employed in the feature engineering module to establish the optimal sequence for packet categorization. The Bootstrapped Optimistic Process for Tree Structure with ANN is used to classify packages into five categories in the classification unit.

A deep BiLSTM blockchain architecture was presented by Alkadi, O., et al. [30], utilizing the datasets from UNSW-NB15, as well as BoT-IoT, to provide security-assisted decentralized ID and privacy-integrated blockchain with smart agreements in IoT systems. This architecture was deployed as a decision-support software, which supports customers and cloud workers to move their data safely, quickly, and reliably.

Mansour, R. F. [31] has developed a powerful blockchain-aided cluster-integrated IDS for the IIoT, known as the BAC-IDS approach. This model employs Hawks Optimization (HHO)-supported clustering to effectively select Cluster Heads (CH) and create clusters as necessary. To ascertain whether intrusions are present in the IIoT environment, Chicken Swarm Optimization (CSO) incorporated with Gated Recurrent Unit (GRU)-aided ID is also employed. The innovative aspect of the suggested study is the development of HHO grouping, and CSO is applied for the optimization of hyperparameter for the IIoT context.

The Interplanetary File System (IPFS)-based storage authenticity and access control model and the Chronological Anticorona Virus Optimization (CACVO-based DRN)-based detecting network intrusions system was created by Saviour, Mariya Princy Antony, et al. [32]. DRN was then utilized to carry out network attack detection. The feature synthesis mechanism, which employs a Deep Belief Network with fusion correlation variables, receives the recorded

data log file from doing this (DBN). The suggested optimization method, CACVO, was recently created by combining the CACVO method, and is utilized to carry out detection mechanisms using DRN once the feature merging is finished.

A unique technique for a network IDS was suggested by Nguyen, Minh Tuan, and Kiseon Kim [33], employing a better feature subsection unswervingly chosen by a genetic algorithm (GA)-incorporated comprehensive search and fuzzy C-means clustering (FCM). By conjoining the GA with 5-fold cross-validation to select the CNN prototypical arrangement, the algorithm determines the bagging classifier and the CNN model as an efficient extractor. To verify performance with the 5-fold CV, the deep feature subsection extracted by the chosen CNN prototypical is fed into the BG classifier. The summary of the literature is analyzed in Table 1.

Table 1. The summary of the related work.

Reference	Dataset	Detection Method	Prevention Method	Key Findings	Limitation/ Recommendation
Retnaswamy, Bharathi [27]	NSL-KDD	MWKF-LSTM with DEDFA	SHA-512 and E^2CC	Low FPR and Detection delay	Accurate detection is not possible due to a high error rate
Pooja, T. S., and Purohit Shrinivasacharya [28]	KDDCUP99 and UNSW-NB15	Bi-directional LSTM	-	99% accuracy is achieved	Poor execution time and large amount of training time
Islabudeen, M., and M. K. Kavitha Devi [29]	NSL-KDD	SA-IDPS	SHA-256	Assistance for a free environment with a high detection rate, FPR, and less energy consumption and delay	Large dataset processing is challenging
Alkadi, O., et al. [30]	UNSW-NB15 and BoT-IoT	BiLSTM	-	Effective in basic real-world scenarios	May not work for extended computations
Mansour, R. F. [31]	NSL-KDD2015 and CICIDS 2017	BAC-IDS	-	Able to prevent end-to-end communication	More false positives compared to other approaches
Saviour, Mariya Princy Antony, et al. [32]	Network dataset	CACVO-based DRN	-	Flexible, in terms of features	The problem of uncertainty still needs to be resolved. Unsuitable for sophisticated applications, especially intrusion detection
Nguyen, Minh Tuan, and Kiseon Kim [33]	NSL-KDD	GA-FCAM-CNN	-	Additionally, the extremely trustworthy validation Performance results attained	IDS speed is a crucial statistic. However, it is extremely low, and any outside source does not update the trust calculation

3. Proposed Methodology

The research methodology is illustrated in Figure 1. This study uses KDDCUP99 and UNSWNB-15; the data goes through the data pre-processing stage, where the data is cleaned and ready to be used in the experiment, and this indicates that the information preparation block is completed. At this stage, the attack type and dataset length are also computed—normalized dataset saved in a CVS file. Train and test sets are separated from the normalized data. The SSO algorithm selects the significant features from the dataset. Using the suggested model in conjunction with an LSTM strengthens and saves time on the network. An LSTM's training process is governed by the "MSE" loss function and log loss function.

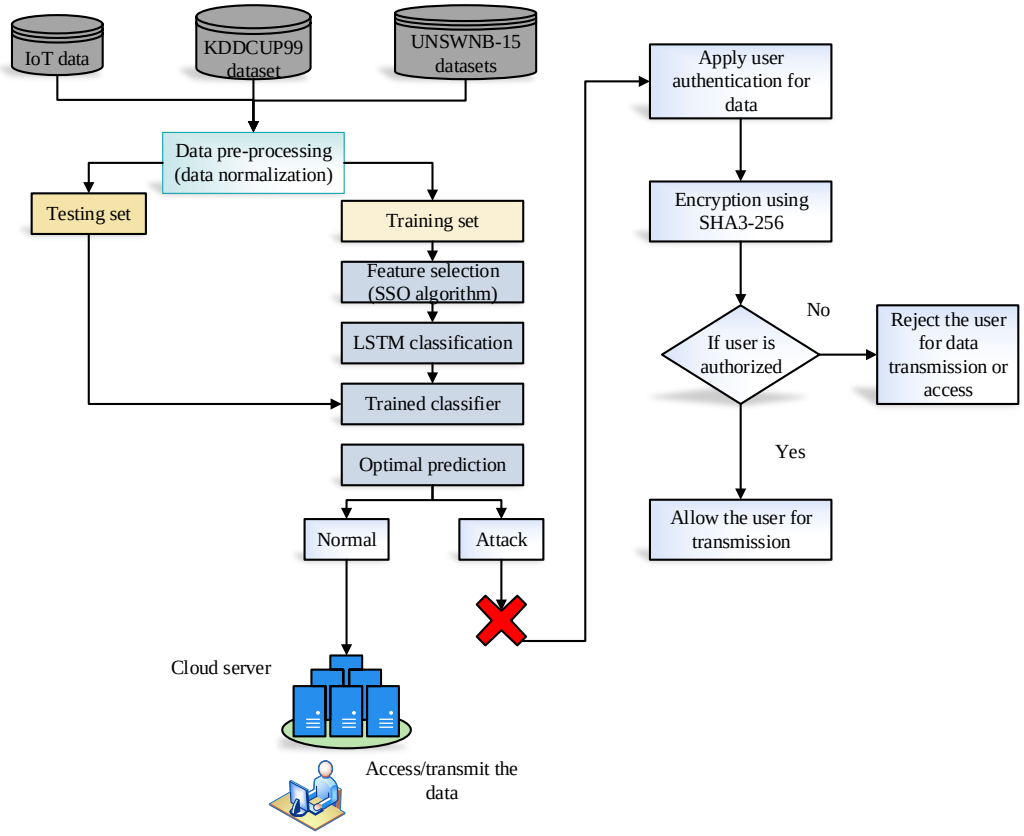


Figure 1. Proposed system flow diagram.

3.1. Data Pre-Processing

After data is gathered during the data collection stage, it must first be administered to estimate the major properties of the KDDCUP99 and UNSWNB-15 datasets. The two main phases of the data pre-processing procedure are data transfer and normalization. Data communication is required, since the trained classifier utilizes only statistical numbers in training and testing phases. Hence, data pre-processing is necessary to convert the non-statistical input into statistical values. Both numerical and non-numerical data are present in the dataset. Furthermore, definite values were allocated for every parameter to convert the non-numeric qualities of the training and test datasets to numeric types. For example, the attributes available in the database contains either discrete or continuous numbers; as a result, they would each have a unique range for the chosen characteristic, making them incomparable. The range of each feature is normalized using the min-max method. Additionally, this made it possible to map the whole [0, 1] range of possible values for each characteristic. To express the min-max normalization, use Equation (1):

$$x_{m,n}^{norm} = \frac{x_{m,n} - \min(x_{:,n})}{\max(x_{:,n}) - \min(x_{:,n})} \tag{1}$$

where $n = 1, 2, \ldots i$ and i are the features count; $m = 1, 2, \ldots j$ and j define the train or test data quantity, $x_{m,n}$ is a parameter assessment y_m of the feature n; $x_{m,n}$ denotes the value of n^{th} attribute; $\min(x_{:,n})$ refers to minimum rate of the column feature; and $\max(x_{:,n})$ indicates maximum rate of the column feature topping map component between (0, 1).

3.2. SSO for Feature Selection

The optimal fitness of the SSO algorithm selects the features from the normalized data. Horses or other animals are herded together by shepherds according to their natural

tendencies for selecting the optimal path in the field. The shepherd herds sheep while riding horses to do this. This conduct led to the formation of the SSO.

Initialization: The mathematical model starts SSO in the solution space with a beginning population parameter that is chosen at random, presented in Equation (2):

$$L_{u,v}^{0} = L_{\min} + rand \times (L_{\max} - L_{\min}); \quad \text{u} = 1, 2, \ldots \text{gandv} = 1, 2, \ldots \text{h} \tag{2}$$

where $L_{\min}$ and $L_{\max}$ indicate the lowermost and supreme archetypal constraint limits, respectively; *rand* is an arbitrary parameter molded between 0 and 1 for every component; g represents individual count in each collection; and h defines the collection count. Additionally, this technique allows for the computation of the total number of community members using Equation (3):

$$vL_p = g \times h \tag{3}$$

Feature matrix: In this method, the starting parameters g of each population are, depending on their objective functions, randomly assigned to the first column of the multi-community conditions of Equation (4). The next parameter g is selected in the same manner as the previous one, and it is arranged in the unit to produce the second column of the multi-community limit in random order. This process is carried out repeatedly h until the subsequent multi-community matrix is produced. The dimension of the matrix is $(g \times h)$; this indicates that the feature matrix contains g communities and that each community contains h members, in which u and v define the index of the community and index of the member within the community.

$$L_p = \begin{bmatrix} L_{1,1} & L_{1,2} & \cdots & L_{1,h} & \cdots & L_{1,h} \\ L_{2,1} & L_{2,2} & \cdots & L_{2,h} & \cdots & L_{2,h} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ L_{u,1} & L_{u,2} & & L_{u,v} & & L_{a,h} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ L_{g,1} & L_{g,2} & \cdots & L_{g,v} & \cdots & L_{g,h} \end{bmatrix} \tag{4}$$

It is important to note that each row of the multi-community constraint indicates each member in the group, with the best members of each group appearing in the top column. Additionally, the individuals in the final phase are the group's weak parameters.

Optimal feature selection: Two criteria are used to create a customized step size for each group member. The first variable $K_{u,v}^{w}$ indicates the possibility of looking into additional regions of the solution space. The second factor $K_{u,v}^{b}$ is the ability to look around previously visited potential solution space regions. The step size's mathematical expression is shown in Equation (5):

$$K_{u,v} = K_{u,v}^{w} + K_{u,v}^{b} \quad \text{a} = 1, 2, \ldots \text{x} \ \ \& \ \ \text{b} = 1, 2, \ldots \text{y} \tag{5}$$

Equations (6) and (7) estimate the worst and more optimal functions of step size for tweaking the parameter:

$$K_{u,v}^{w} = \chi \times rand_1 \times (L_{u,w} - L_{u,b}) \tag{6}$$

$$K_{u,v}^{b} = \zeta \times rand_2 \times (L_{u,b} - L_{u,v}) \tag{7}$$

In comparison to $L_{u,v}$, $rand_1$ and $rand_2$ are random parameters, with each component having a value between 0 and 1; $L_{u,b}$ and $L_{u,w}$ are the better and worst parameters in terms of the objective function value. It is worth noting that the m^{th} community's initial parameter $L_{u,1}$ lacks an affiliate that is grander to it. As an effect, $K_{u,v}^{b}$ has the same value as 0. As a consequence, the collection's last features do not have a poorer limit than itself. As a result,

$K_{u,v}^{w}$ is also 0. Besides, χ and ζ are the variables that influence both exploration as well as exploitation. The definitions for the variables are represented in Equations (8) and (9):

$$\chi = \chi_0 - \chi_0 \times n; \quad n = \frac{\text{Value of iteration}}{\text{Overall rate of iteration}} \tag{8}$$

$$\zeta = \zeta_0 + (\zeta_{\max} - \zeta_0) \times n \tag{9}$$

It is self-evident that, as the number of iterations n grows, the value χ increases and lowers ζ. As a result, exploration lags while exploitation lifts.

Feature updating: The new parameter $L_{u,v}$ is computed using Equation (10) and the previous step. The parameter will then be adjusted if $L_{u,v}$ (its current objective function rate) is not less than its previous rate:

$$L_{new\ u,v} = L_{u,v} + K_{u,v} \tag{10}$$

Once the maximum number of iterations has been achieved, the optimization process will conclude. If not, it goes through a second round of repetitions of step one. The Algorithm 1 contains the SSO algorithm's pseudocode for feature selection.

Algorithm 1. SSO pseudocode for feature selection

Start

1. Initialize the objective and define the system parameter
2. Estimate the community of features randomly
3. Sort the community of features in ascending order based on the goal function
4. Create the subset sort:
   - 4.1 Compute step size matrix
   - 4.2 Compute a new community of matrix
5. Estimate the new community of features
6. Provide the replacement method between the new and updated community of features
7. Report the best features

End

3.3. LSTM Classification

Typically, Deep Packet Inspection (DPI) techniques involve analyzing the content of packets to classify traffic. DPI can provide detailed information about the application or protocol used but may need more performance limitations to be effective against encrypted traffic. However, LSTM-based approaches can complement DPI by focusing on the payload content and capturing patterns that may not be easily identifiable through DPI alone, since the payload of a packet can be compared via DPI to a set of criteria, which are typically presented in string format. However, using such format rules comes with severe restrictions, including a lack of expressiveness and an inability to handle a variety of complicated services [34]. LSTM algorithms are strong instruments that aid computer understanding and prediction of complicated data. LSTM aims to solve the long-term reliance issue by collecting data over a long time. By integrating a memory cell, it is feasible to track these connections across the whole series. An LSTM is composed of three gates: an input, an output, and a forget. The input gate defines how much fresh data will be placed into the memory, the output gate determines if the current value in the cell subsidizes the output, and the forget gate chooses whether to keep or delete previous data. Figure 2 depicts the LSTM approach's organizational structure.

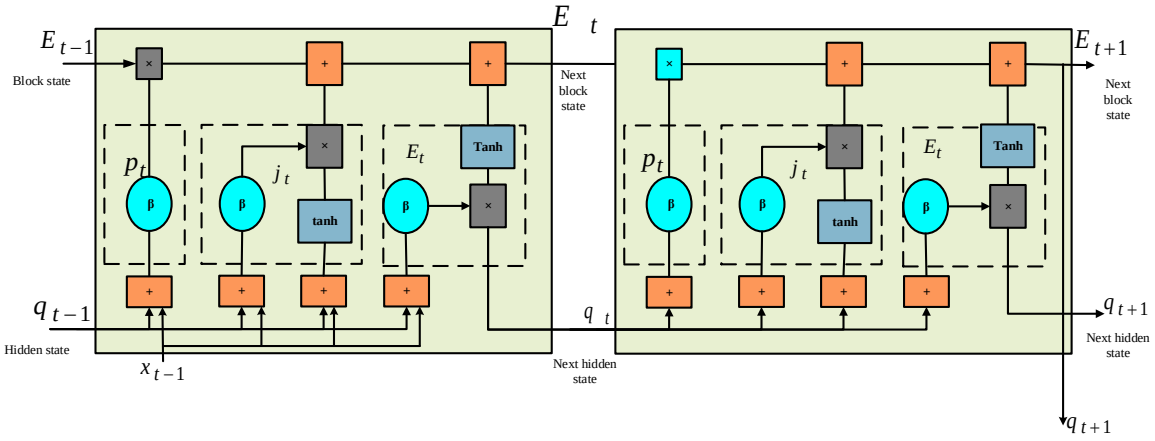


Figure 2. Proposed LSTM architectures.

An LSTM network's fundamental unit of memory is the cell. The cell state and the concealed state are the two states that are passed to the subsequent cell. The primary data flow chain that enables the data to move forward essentially unmodified is the cell state. However, certain linear changes could take place. Sigmoid activation gates can be used to add or delete data from the cell state [35]. A gate is comparable to a layer or a group of independently weighted matrix operations. Since gates are used to regulate the learning process, LSTMs with sigmoid functions are designed to get over the long-term reliance problem. Building an LSTM network begins with identifying data that is superfluous and will be excluded from the cell. To determine if data requirements are to be disconnected from the memory of LSTM, the forget gate uses the sigmoid function. The choice is determined based on the values of q_{t-1},x_t. The outcome of this gate p_t has a rate between 0 as well as 1, where 0 denotes entirely throwing out each rate and 1 denotes retaining all whole standards. Equation (11) is used to calculate the result:

$$p_t = \sigma\left(W_p[q_{t-1}, x_t] + b_p\right) \tag{11}$$

where the concealed layer has a bias b_p, the forget gate vector exists. The input vector weighted by the hidden state is W_p. The gate of input (x_t) decides whether or not to augment the new information to the LSTM memory. The "sigmoid" layer and the "tanh" layer are the two layers included in this gate. The responsibility of the tanh layer is to provide a path of the new potential parameter E_t that will be inserted into the LSTM memory E_{t-1}, while responsibility of the sigmoid layer is to decide which variables are updated. Equations (12) and (14) are used to calculate the output of these layers:

$$j_t = \sigma\left(W_j[q_{t-1}, x_t] + b_p\right) \tag{12}$$

$$S_t = \tanh(W_S[q_{t-1}, x_t] + b_S) \tag{13}$$

$$\hat{E}_t = \text{tanhtanh}(W_E[q_{t-1}, x_t] + b_E) \tag{14}$$

where x_t is the input gate vector. The bias vectors are the hidden layer's weight, being b_p and W_j. Equation (15)'s potential vector indicates that, after merging two layers, the LSTM memory sends an update in which the current value is replaced by doubling the previous value (i.e., by introducing a new variable $\hat{E}_t$):

$$E_t = E_{t-1}q_t + S_t j_t \tag{15}$$

Here, W and b are used to signify the weight matrices as well as the bias of the cell condition, respectively, and E_{t-1} and E_t are used to designate the cell conditions at time

$t-1$ and t. Although they are a filtered form, the output values q_t in the final step are dependent on the output cell state y_t. The initial choice of which basics of the cell state E_t are output is made by a sigmoid layer. The output of the sigmoid gate y_t, which has a value between 1 and 1, is then multiplied by the new principles, which the tanh layer creates from the cell. To choose which area of the LSTM memory is allocated to the outcome, the output gate first uses a sigmoid. A non-linear function of tanh is used, as a result, to get a number between -1 and 1. Of course, the output of the sigmoid layer is multiplied by the outcome. Equations (16) and (17) are used to compute the output:

$$y_t = \tanh\left(W_y[q_{t-1}, x_t] + b_y\right) \tag{16}$$

$$q_t = y_t \tanh(E_t) \tag{17}$$

Here, W_y and b_y stand for the output gate's weight and bias matrices, respectively. Finally, the optimal type of malicious traffic attack or normal condition is predicted by the developed model.

4. Intrusion Prevention

To prevent hostile nodes from entering the network, intrusion prevention is crucial. We suggested One-Way Hash Chain Function utilizing SHA3-256 for intrusion protection. It is useful for authentication by generating hash values and may be utilized in a variety of network security applications. To disrupt the IDS or attempt to establish contact between valid nodes to get data packets, intruders may create false identities from legitimate nodes. As a result, SHA3-256 worked well with the security system. However, always processing the complete message at once prevents the vulnerability, which necessitates the creation of a temporary buffer. Moreover, the attacks by collision are possible. Therefore, the sponge construction of the Keccak function is applied in the developed SHA3-256 hash function. A source bit stream of any length may be used with this method, which has a fixed internal configuration and generates an outcome bit stream of any length that is desired. The structure of SHA3-256 is provided in Figure 3.

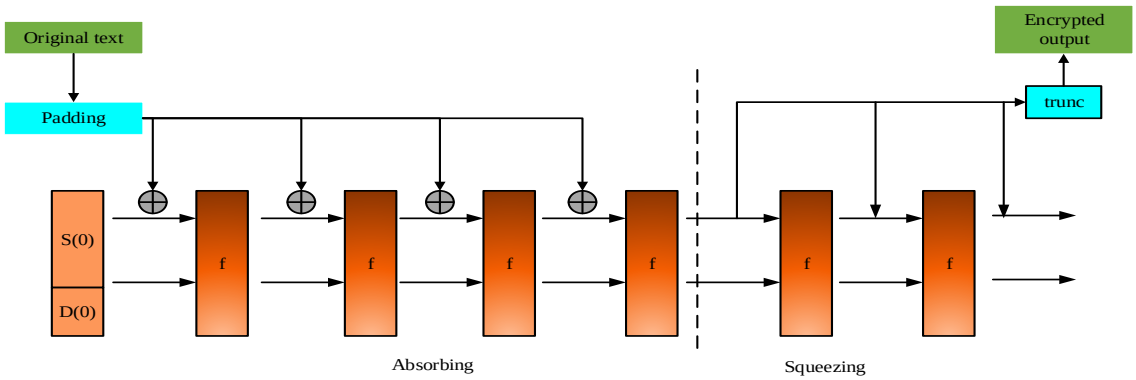


Figure 3. Structure of SHA3-256 security function.

Python's "hashlib" module is used to generate the cryptographic hash value, and SHA3 256's 32-bit digest size is employed. The sponge design used by SHA-3 allows data to be "absorbed" into the surface and subsequently "squeezed" out. In the absorption phase, blocks that include data messages are XORed into a portion of the condition, and the resulting condition is then completely altered using a permutation function f. In the "squeeze" phase, the condition transformation function f substitutes with interpretation output blocks from the equal subsection of the condition. The dimension of the condition's writeable and readable portions is referred to as the "rate (S)", and the dimension of the remaining portions is referred to as the "capacity (D)". The security of the system is evaluated by capacity. Partial of the volume is the greatest degree of security. Add padding

to the initial message here. In this manner, the total length is a precise multiple of the hash function's rate. Given that we have chosen SHA3-256 in this instance, it must be a multiple of 1088 bits. The "absorption" and "squeezing" activities are the two primary categories of the SHA3-256 procedure. We have the capacity D and the absorbent design, which produces a bit string y of length m, assuming an input bit string x, a padding task P, a permutation task f that uses bit blocks of size k, a rate S, and an output length m:

$$y = sponge[f, P, D](x, m) \tag{18}$$

By padding the input x with the padding function, a padded bit string R is assumed with a length that may be divided by D (such that is $n = len(R)/D$ an integer). To achieve this, the input data, denoted as "R", is padded using a padding function. This padding ensures that the length of the padded bit string "R" is divisible by the hash function's rate (capacity). Divide R into n successive D-bit parts, $D_0, \ldots D_{n-1}$. Start computing the hash value by absorbing the bloated message values. The padded data is divided into blocks of a set size. Following that, each block goes through 24 rounds of five-operation permutation. A 1600-bit internal data size is all we are left with.

Initialize the condition with a string of k zero bits, include the input in the condition for each R_i block, extend it at the end by a string of zero bits D to make a condition of k length, XOR it with the condition, f, permute the result using the block, and then create the new condition. To get the hash value, "squeeze" is another option. The message is derived from this (squeezed out). Then, 1600 bits, acquired during the absorption operation, are divided based on the corresponding rate and capacity. Initialize f to the blank string. Although f's length is fewer than k, the first S bits of condition are appended to y. Apply f on condition to produce a new condition if y is still fewer than D-bits long. Then, Convert y to k-bits. Create the final hashing result. The 1088 bites are finally reduced to the first 256 bits. The hash digest of the entire message is contained in the extracted value of 256 bits. The foremost bits of the condition are the required hash in SHA3-256 since D is bigger than k, which eliminates the essential for extra block permutations in the stage of squeezing.

The hash algorithm SHA3-256 is applied to the hash value of the data and added as a feature to the data. The Homomorphic Encryption (HE) method is used to maintain data privacy when the hash value has been included in the entirety of the data, and the encrypted information is then sent to cloud storage for analysis. The cloud storage is used to recover encrypted data. To get the original data that was outsourced, the HE is given to the data. To verify the integrity of the data, the subsequent hash value is calculated and compared to the first hash value. The suggested methodology begins to protect and transmit the data if the hash values match; otherwise, it stops working.

5. Results and Discussion

In this section, we examine the tests that were done to measure the efficiency of the projected method. Here, the proposed model's results for IDS and encryption are reviewed, and the model's effectiveness is illustrated by a comparison to other current models that are in use. The suggested model is developed using the CloudSim simulator and the working environment of Python on a computer running Windows 10 with an Intel i7 CPU and 8 GB of RAM.

5.1. Dataset Description

In this work, two kinds of datasets are mainly used for testing: KDDCUP99 and UNSW-NB15.

KDDCUP99: This dataset was modeled for ID in an expanded form. The following four attack types are given in the KDDCUP99 Dataset as Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), and Probing. There are 41 characteristics in total, 38 of which are numeric, and 3 of which are not (protocol type, service type, and fag). There are also fundamental traffic features (23–41), content features (11–22), features (1–10), and a class label for each item.

UNSW-NB15: There is a massive amount of network traffic in the UNSW-NB15 dataset, and about two million records of both anomalous and regular connections. This dataset, which reflects real-world situations, is high-dimensional, with 49 attributes for each link record and a mean velocity of 5–10 MBs. The UNSW-NB15 dataset, one of the most widely used benchmark datasets for testing IDS, is old but still contains patterns of nine contemporary attack types, including Backdoors, Fuzzers Inquiry, Denial of Service (DoS), Shellcode, Exploits, Generic, Reconnaissance, and Worms.

The proposed work was evaluated with these two IDS datasets. On the other hand, the real-time data collection involves a network of IoT devices that interact with each other, possibly under various conditions or scenarios. These conditions could include different network loads, device configurations, and types of traffic. The data collected from this IoT setup will likely include information about network traffic patterns, communication between devices, and potential anomalies or security events. The proposed technique follows the same procedure for the real-time data to predict and mitigate the attacks. Its ability of differentiating malicious and normal network traffic enables to be more efficient in real-time conditions.

5.2. Validation Parameters

The model's results are analyzed by estimating parameters such as recall, accuracy, specificity, f-measure, false positive rate (FPR), precision, log loss, and FNR. These parameters provide the efficiency of the IDS methods in identifying the malicious network traffic. Furthermore, to manifest the influence of the classification process, a wide range of indicators are typically utilized in IDS research.

Accuracy: It is a crucial performance indicator for assessing IDS, well-defined as the entire amount of data that was successfully classified out of all the packets transmitted. The formula for accuracy calculation is represented in Equation (19):

$$\text{Accuracy} = \frac{\hat{\text{T}}\hat{\text{N}} + \hat{\text{T}}\hat{\text{P}}}{\hat{\text{T}}\hat{\text{N}} + \hat{\text{T}}\hat{\text{P}} + \hat{\text{F}}\hat{\text{N}} + \hat{\text{F}}\hat{\text{P}}} \tag{19}$$

Recall: It is known as detection percent and is indicated as ratio of instances that have been legitimately tested to all positive trials. A critical IDS metric and detection rate demonstrate how well the model can recognize attacks, mathematically expressed in Equation (20):

$$\text{Recall/Sensitivity} = \frac{\hat{\text{T}}\hat{\text{P}}}{\hat{\text{F}}\hat{\text{N}} + \hat{\text{T}}\hat{\text{P}}} \tag{20}$$

Precision: The confidence in attack detection is measured by the ratio of actual positive models to predicted positive models. The formula for precision estimation is defined in Equation (21):

$$\text{Precision} = \frac{\hat{\text{T}}\hat{\text{P}}}{\hat{\text{F}}\hat{\text{P}} + \hat{\text{T}}\hat{\text{P}}} \tag{21}$$

F-measure: F-measure is estimated based on the harmonic mean of recall and accuracy. The f-measure formula is represented in Equation (22):

$$\text{F} - \text{measure} = \frac{2 * \text{Recal} * \text{Precision}}{\text{Recall} + \text{Precision}} \tag{22}$$

False Negative Rate (*FNR*): The FNR is the proportion of false negative samples to all positives. The FNR is often referred to as the assault detection missed alarm rate and is expressed in Equation (23):

$$\text{F}NR = \frac{\hat{\text{F}}\hat{\text{N}}}{\hat{\text{T}}\hat{\text{P}} + \hat{\text{F}}\hat{\text{N}}} \tag{23}$$

FPR: It is described as the ratio of models that tested falsely positive to those that were anticipated to test positive. In attack detection, the FPR, sometimes referred to as the false alarm proportion, is calculated using Equation (24):

$$FNR = \frac{\hat{F}\hat{P}}{\hat{T}\hat{P} + \hat{F}\hat{P}} \tag{24}$$

Log-Loss Score: The logarithmic loss reveals how well a forecast probability matches the true value or actual value. Log loss is the averaged negative value of the adjusted predicted probability for each case. The Log-Loss score calculation is expressed in Equation (25):

$$LogLoss = -\frac{1}{N}\sum_{j=1}^{N}(\log(P_j)) \tag{25}$$

where $\hat{T}\hat{P}$, $\hat{T}\hat{N}$, $\hat{F}\hat{P}$, and $\hat{F}\hat{N}$ are used to indicate the true positives, true negatives, false positives, and false negatives, respectively. Malicious traffic data are typically recognized as positives and normal data as negatives, since the objective is to determine the attacks. Metrics such as recall, accuracy, FNR, and FPR are widely used in attack detection.

5.3. Performance Evaluation

In this research, the KDDCUP99 and UNSW-NB15 datasets are taken for the validation. The raw data from the database is pre-processed for removing the noise and missing values. Further, the SSO algorithm is applied for the feature selection. Some aspects have little bearing on determining whether or not the information and traffic behavior is normal. We employ 90 characteristics instead of the original 89 features, since such features, including the timestamp feature and IP addresses, do not aid in training the neuron to identify mistakes and intrusions. The performance measure's value of selected features for both datasets are analyzed and equated with the earlier models, and are provided in Table 2.

Table 2. Validation of metrices for feature selection.

Methods	Dataset	Features	Accuracy	Precision	Recall	F-Measure	FPR	FNR	Communication Delay (s)
DEDFA [27]	KDDCUP99	35	67	68	67	86	0.005	1.05	30.7
	UNSW-NB15	45	64	64	62	64	0.045	4.12	55.8
BOAT [28]	99KDDCUP99	41	81	81	64	80	0.009	1.054	31.5
	UNSW-NB15	35	75	82	67	81	0.008	0.15	24.12
DFPFS [29]	KDDCUP99	42	76	81	68	64	0.00564	0.16	22.6
	UNSW-NB15	42	74	76	64	61	0.005	0.2	23.0
Feature fusion [32]	KDDCUP99	41	71	81	69	78	0.005	0.12	21.8
	UNSW-NB15	46	82	92	75	90	0.071	0.615	27.8
GA [33]	KDDCUP99	32	91	81	79	56	0.054	0.6	27.2
	UNSW-NB15	38	72	89	82	92	0.01	0.05	21.9
Proposed	KDDCUP99	40	99.92	98	95	98	0.09	0.015	7.23
	UNSW-NB15	49	99.91	97.4	96	94.5	0.091	0.011	6.02

Further, the selected features are applied to the LSTM approach, whether the transmission process have attacked or not, independent of what types of attack are obtained. There are two different kinds of categorized classes; one is normal and others are malicious traffic attacks, such as Denial of Service (DoS), User to Root (U2R), Probe, Remote to Local (R2L), Backdoors, Fuzzers Inquiry, Shellcode, Exploits, Reconnaissance, and Worms. Consequently, the SHA3-256 with HE method is applied for secured data transmission. The execution of the projected method is compared with the earlier methods such as

MWKF-LSTM with DEDFA [27], SA-IDPS [29], BiLSTM [30], and GA-FCAM-CNN [33] regarding the evaluation metrics, including accuracy, FPR, FNR, encryption time, and decryption time.

In the presented strategy, adaptive learning was utilized for training the LSTM for classifying the malicious and normal network traffic. Adaptive learning indicates the capacity of the design to continuously update itself based on new data. In the presented work, adaptive learning helps the system to evolve and improve its accuracy as it encounters new and previously unseen attack patterns, thereby increasing its efficiency in attack prediction. To manifest the learning efficiency of the designed method, it is equated with different learning algorithms, such as reinforcement learning (RL) [36], adaptive reinforcement learning (ARL) [37], unsupervised learning (UL) [38], and semi-supervised learning (SL) [39]. Figure 4 presents the learning rates of different techniques. Figure 5a,b provide a graphical presentation of the training accuracy, as well as a loss comparison for the two (KDDCUP99 and UNSW-NB15) databases. The suggested system's ability to continue operating after 25 epochs of training accuracy and 50 epochs of training loss is proven. This graph is displayed between focused log loss and epochs. Comparing the suggested system's training performance to that of older models reveals that the new approach has a greater training accuracy and a very low loss value.

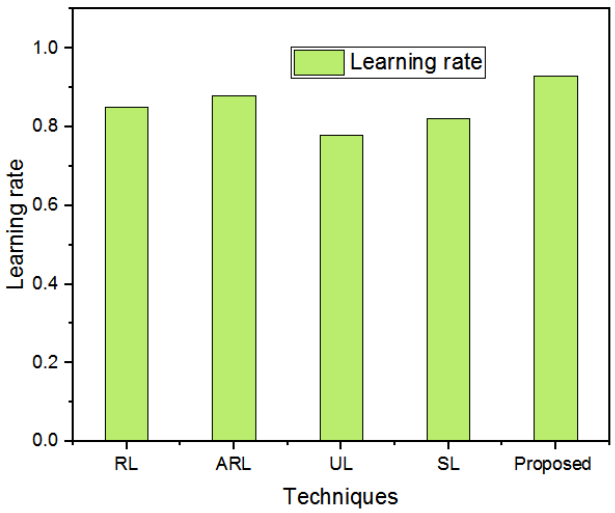


Figure 4. Learning rate comparison.

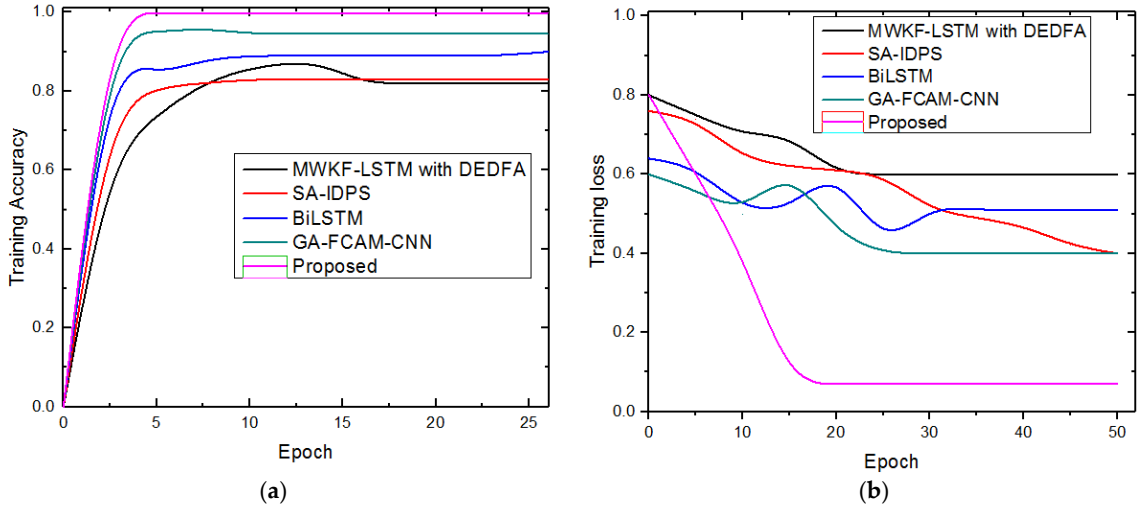
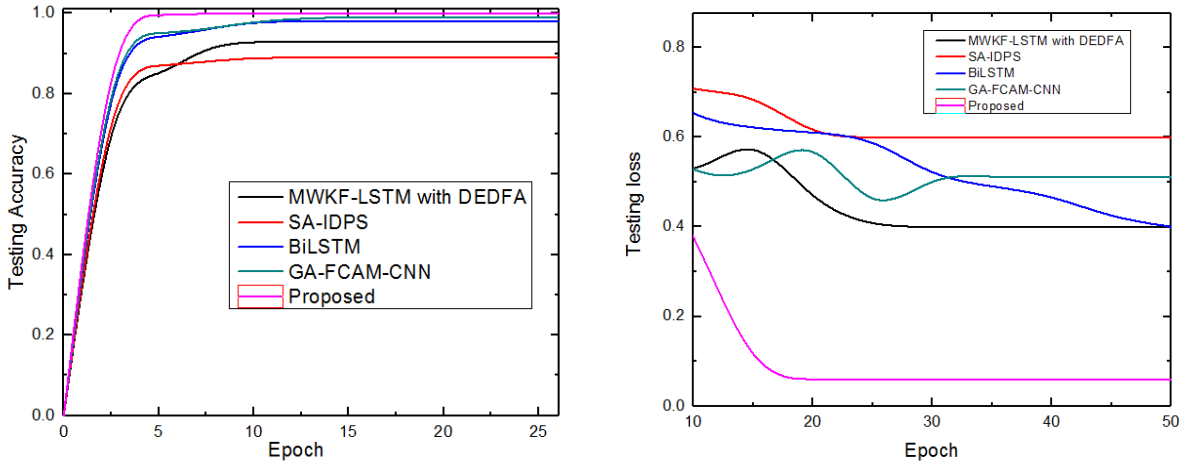


Figure 5. (**a**,**b**) Comparative analysis of training accuracy and loss for both datasets.

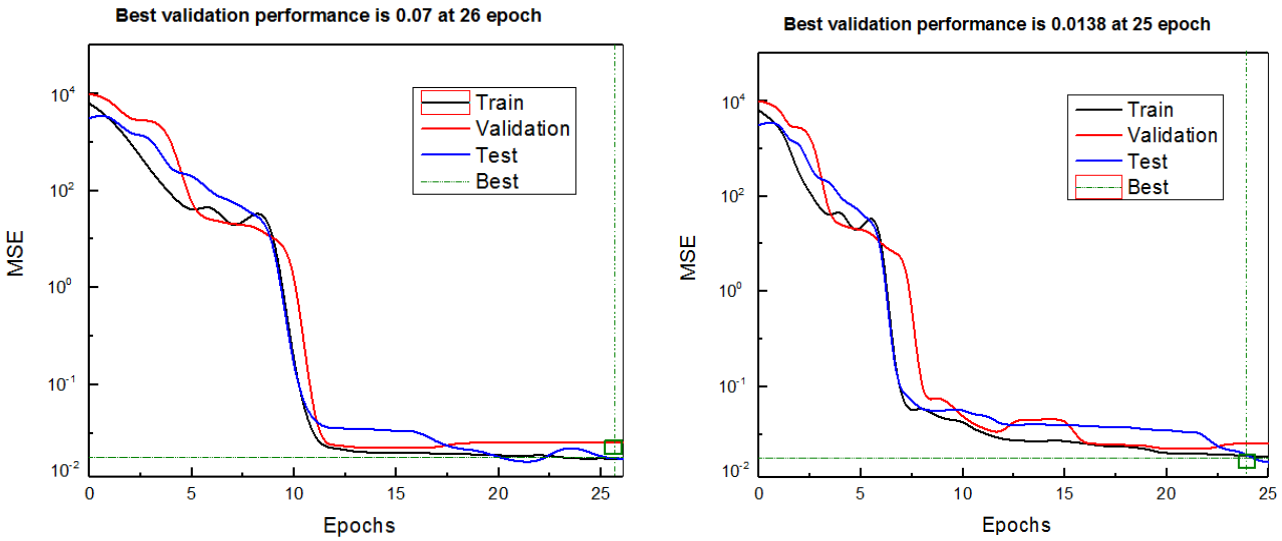
Figure 6a,b provides the comparative analysis of testing accuracy and loss for both datasets. The suggested system's ability to continue operating after 25 epochs of training accuracy and 50 epochs of training loss is proven. This graph is displayed between focused log loss and epochs. The proposed approach has achieved higher testing accuracy with very low loss function over the conventional methods.



(**a**) (**b**)

Figure 6. (**a**,**b**) Comparative analysis of train and test loss for both KDDCUP99 and UNSW-NB15 datasets.

The MSE value of the proposed method is validated for both datasets, as illustrated in Figure 7. It is utilized to evaluate, train, and test performance records. At epoch 26, the KDDCUP99 dataset's validation performance is at its best (0.07). The training epochs with the lowest error yield the highest performance; nevertheless, error reduction frequently starts after validation. As a result, epoch 25 (0.0138) has the best validation performance for the UNSW-NB15 datasets.



(**a**) (**b**)

Figure 7. (**a**,**b**) Performance analysis of the proposed method for KDDCUP99 and UNSW-NB15 datasets.

The evaluation metrics of accuracy, recall, precision, F-measure, FPR, and FNR are estimated for proposed and conventional methods. The graphical representation of those metrics for the KDDCUP99 dataset is portrayed in Figure 8a–f. The investigation displays that the designed method has achieved greater performances regarding accuracy, precision, recall, etc., than the conventional methods.

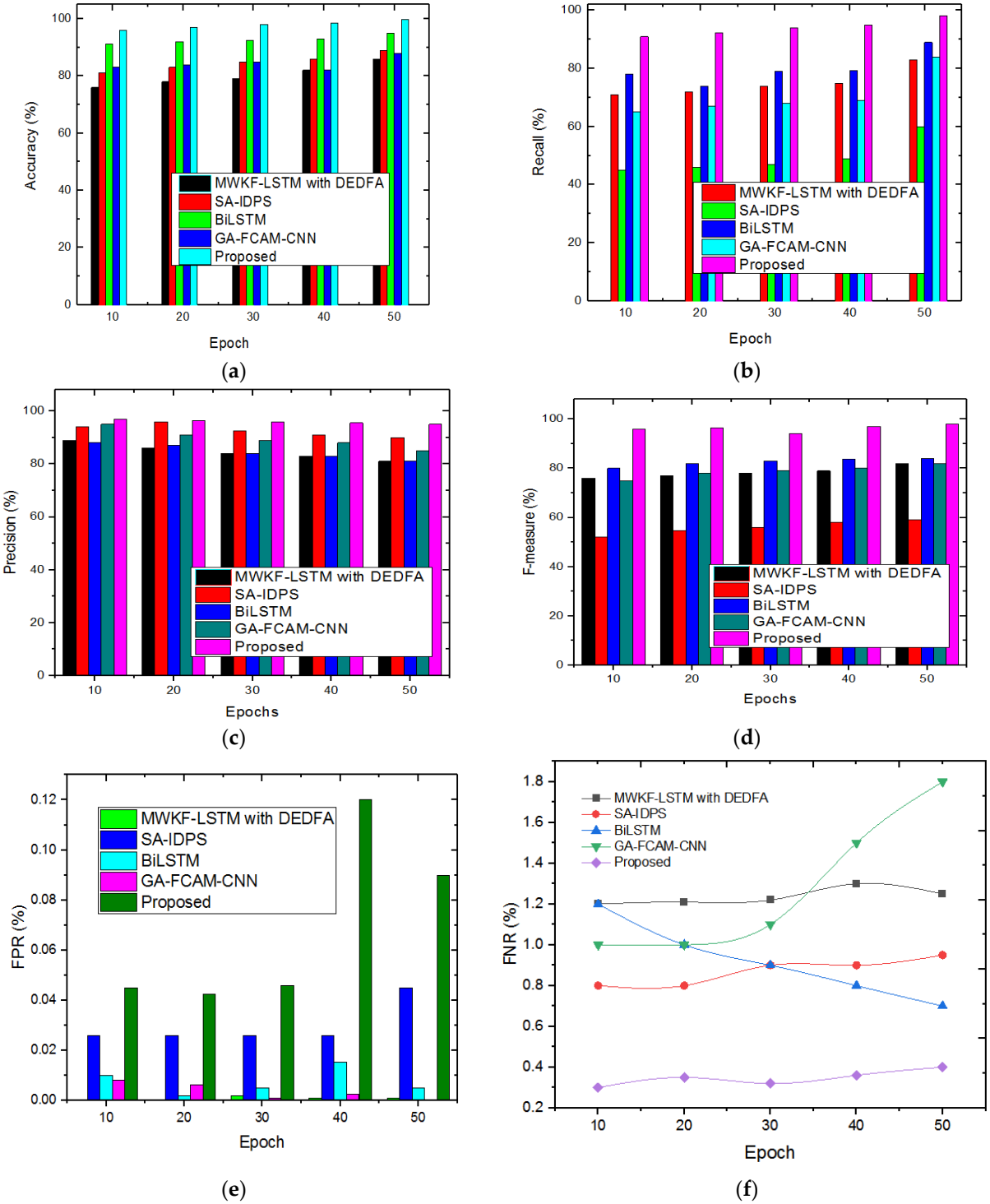


Figure 8. (**a**–**f**) Accuracy, recall, precision, F-measure, FPR, FNR for KDDCUP99 dataset.

Figure 9a–f shows a graphical depiction of those measures for KDDCUP99 database (Figure 9a–f). The study demonstrates that the designed method outperformed the traditional frameworks with higher accuracy, precision, recall, F-measure, FPR, and lesser FNR value.

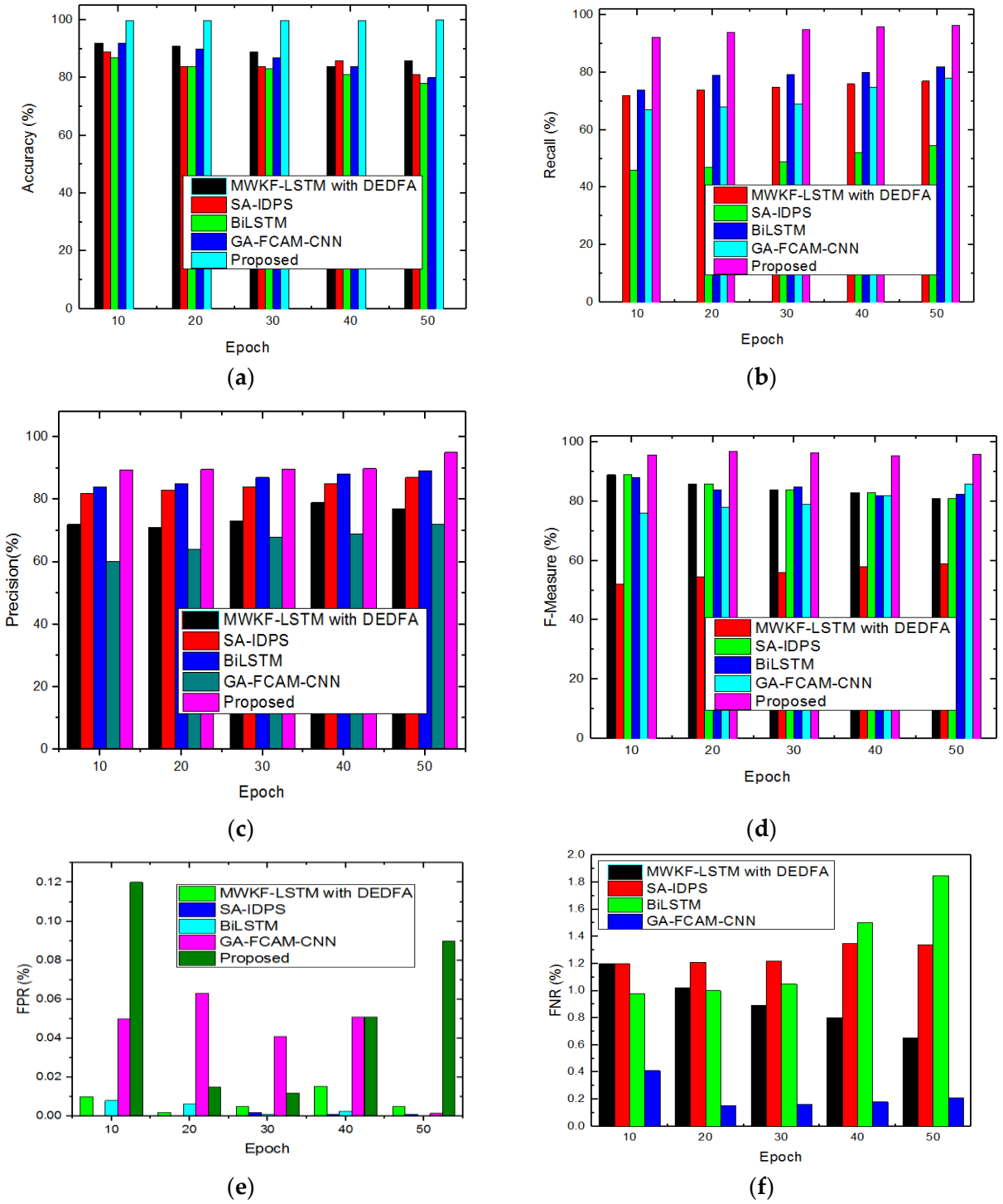


Figure 9. (**a**–**f**) Accuracy, recall, precision, F-measure, FPR, FNR for UNSW-NB15 dataset.

Figure 10a,b provides the communication delay incurred by different techniques for the KDDCUP99 and UNSW-NB15 datasets. The communication delay determines the time taken by the system for transmitting the signal from the sender to the receiver. The comparative study determines that the proposed approach has achieved less communication delay compared to the other tradition methods.

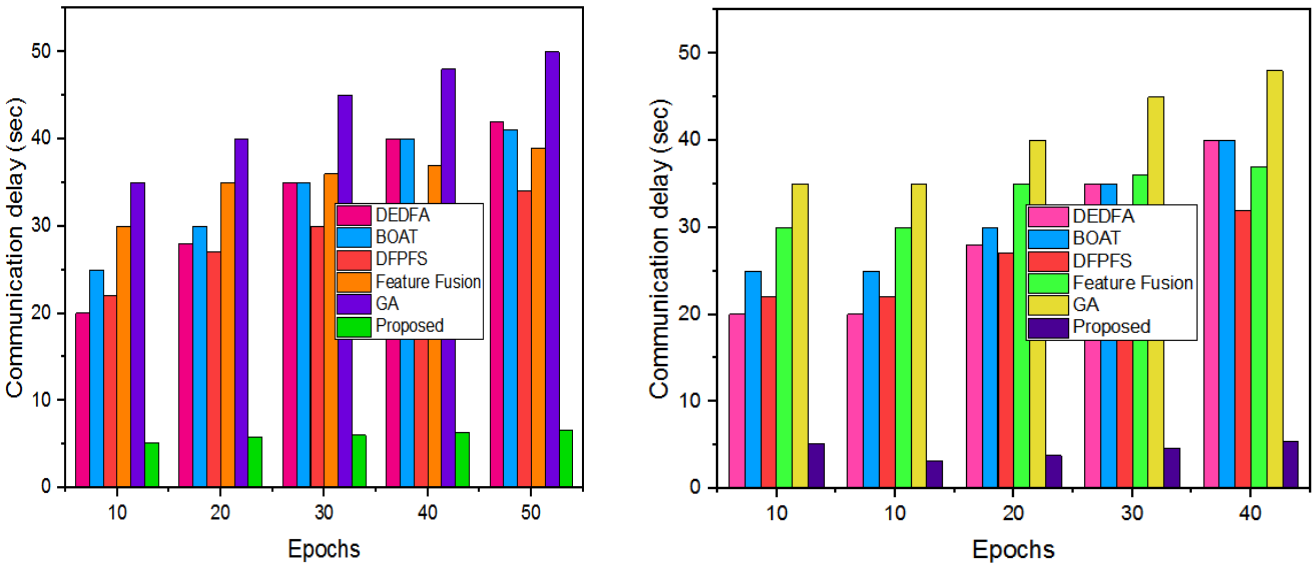


Figure 10. Communication delay: (**a**) KDDCUP99 dataset, (**b**) UNSW-NB15 dataset.

The detection rate efficiency of the designed framework for categorizing the traffic at different packet levels is compared with the conventional models, portrayed in Figure 11a,b. The developed method has achieved a higher detection rate than the other methods.

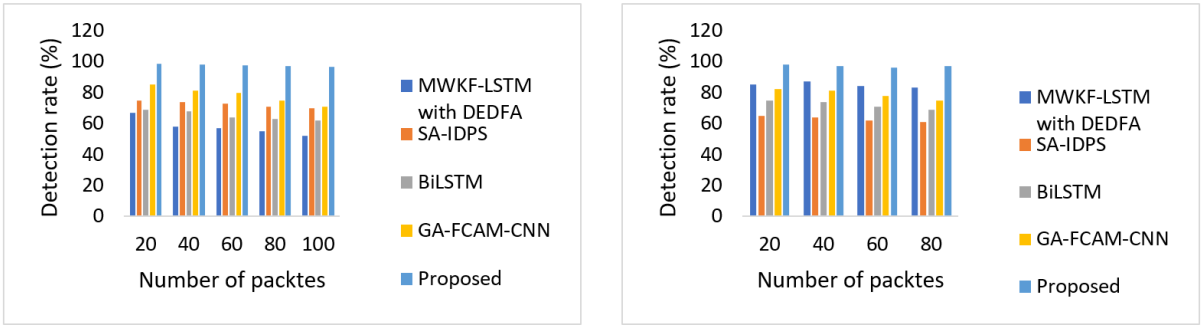
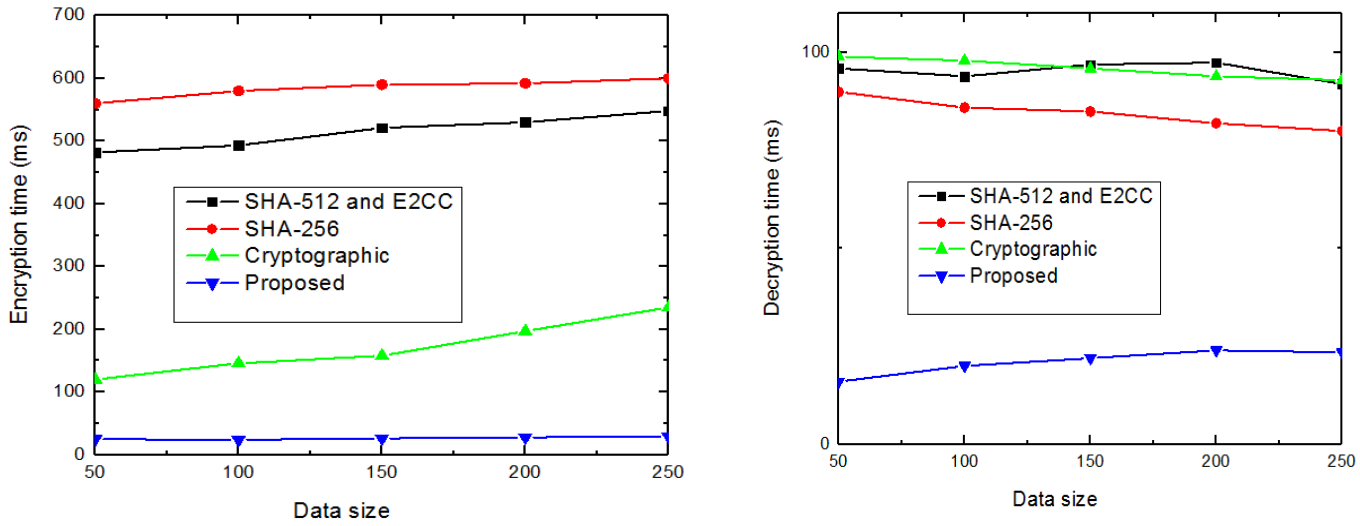


Figure 11. (**a**,**b**). Analysis of detection rate at packet level for the KDDCUP99 dataset and UNSW-NB15 dataset.

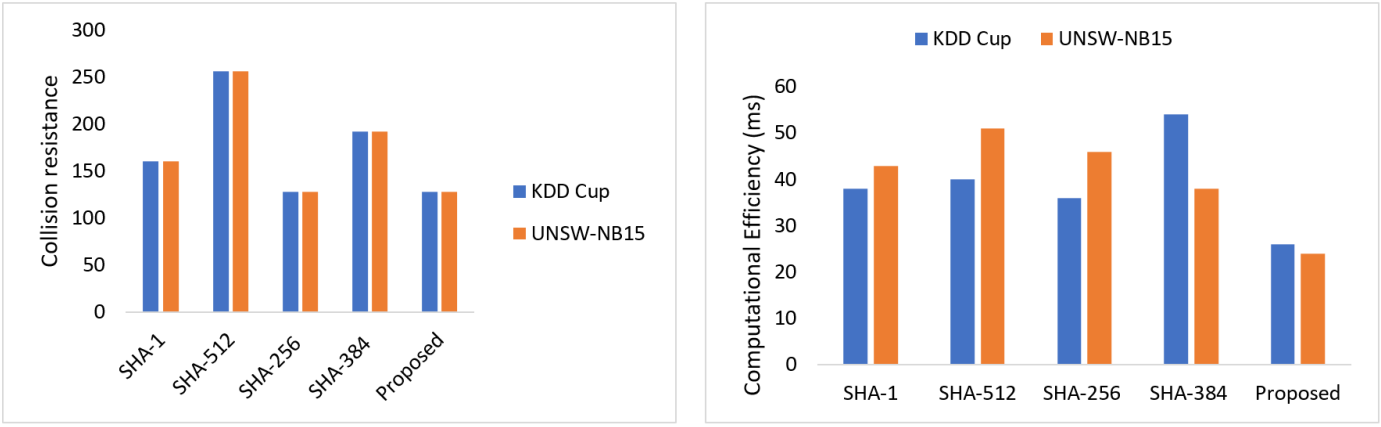
Furthermore, the encryption and decryption time of the presented approach of a SHA3-256-based single-way hash function is compared with the earlier methods, as displayed in Figure 12a,b. The comparative performance describes that the presented methodology has achieved far less encryption and decryption time than the traditional methods.



(**a**) (**b**)

Figure 12. (**a**,**b**). Validation of encryption and decryption time of security algorithm for the KDD-CUP99 dataset and UNSW-NB15 dataset.

Furthermore, the collision resistance and computational efficiency of the proposed a SHA3-256-based single-way hash function is compared with the earlier SHA-1, SHA-512, SHA-256, and SHA-384 methods, portrayed in Figure 13a,b. The results demonstrate that the designed approach achieved very high collision resistance and computational efficiency as compared to the traditional methods. As a result, the proposed SHA3-256-HE is strengthened and becomes more resistant to unidentified attackers. According to the performance investigation, all SHA3-256-HE variations offer higher clock rates per byte than SHA-2 kinds. This result was supported by the SHA3-256-HE internal structure, which makes it more secure because both the MAC and the hash are present, but SHA-2 is not. SHA3-256-HE is still the greatest option for providing security and data integrity, despite having more cycles per byte. SHA3-256-HE's adaptable structure enables it to perform as well as non-anonymity, a non-re and secrecy from all possible attacker configurations. Moreover, the analysis shows that the proposed security algorithm is computationally efficient.

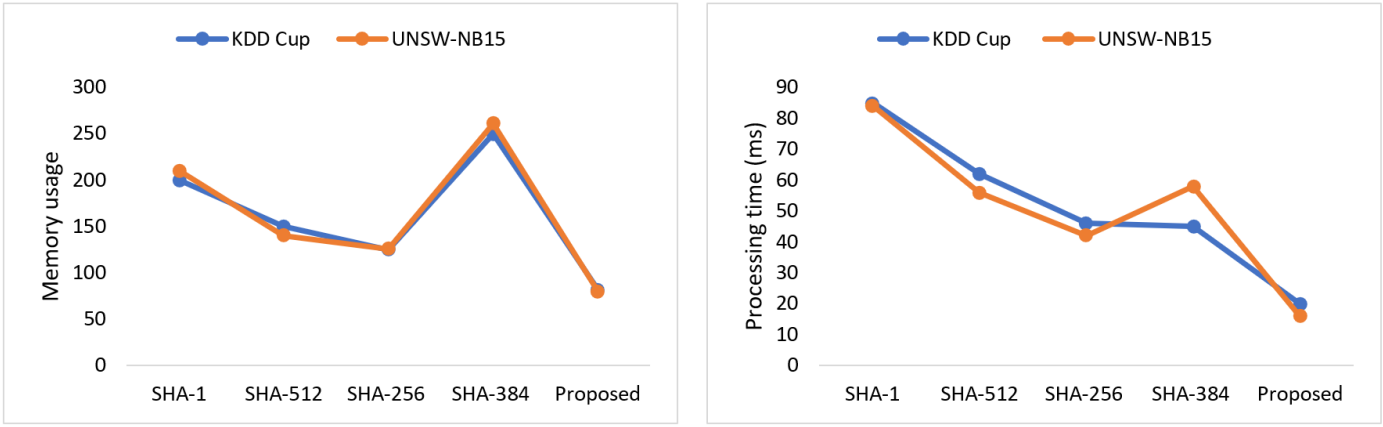


(**a**) (**b**)

Figure 13. (**a**,**b**). Validation of collision resistance and computational efficiency of security algorithm for the KDDCUP99 dataset and UNSW-NB15 dataset.

Due to its low energy consumption, collision resistance, and computational efficiency, SHA3-256 has demonstrated improved performance with small key sizes and is suitable with devices that have restricted resources. Furthermore, the memory usage and processing time of the developed SHA3-256 method is compared with the earlier models when applied to large-scale IoT networks with high data traffic, as demonstrated in Figure 13a,b.

Figure 14a compares how different strategies use memory in this regard. The designed SHA3-256 single-way hashing method was determined to be the best with the least amount of memory needed for processing.



(**a**) (**b**)

Figure 14. (**a**,**b**). Validation of memory usage and processing time of security algorithm for the KDDCUP99 dataset and UNSW-NB15 dataset.

Furthermore, the execution time of the proposed technique was analyzed to determine how much time the proposed technique takes for predicting and mitigating attacks in the IoT environment. Table 3 provides the overall time consumed by the developed framework for KDDCUP99 and UNSW-NB15 datasets. The developed model consumed 4.3 ms and 2.8 ms, respectively, for predicting and mitigating attacks in the KDDCUP99 dataset, while the designed approach attained 3.9 ms and 2.5 ms, respectively, for predicting and mitigating attacks in the UNSW-NB15 dataset.

Table 3. Execution time analysis.

Tasks	Execution Time (ms)	
	KDDCUP99	**UNSWNB-15**
Data Pre-processing	2.1	1.8
Feature Extraction (SSO)	3.5	3.2
LSTM Training	10.2	8.7
Hash Function Calculation	1.2	1.1
Attack Prediction	4.3	3.9
Attack Mitigation	2.8	2.5

5.4. Discussion

Table 4 provides thorough comparison results for the KDDCUP99 and UNSW-NB15 database values. The earlier MWKF-LSTM with DEDFA [26] method used a MWKF model in an LSTM approach with DEDFA feature selection function, yet the optimal solution was not reached due to the complexity and computational burden of this work. In SA-IDPS [28] research, the smart approach is developed, but only limited categories are applied for the intrusion detection and prevention. The BiLSTM [29] for intrusion detection takes more time for training, so the attackers can easily hack the data. Using the GA-FCAM-CNN [32] approach, the trust calculation is not updated by any outside sources and is quite low. The

research demonstrates that the suggested strategy outperformed the standard strategies for both datasets. The proposed method has used SSO for optimal feature selection and an LSTM approach for intrusion detection; also, the data has been secured using the SHA3-256 algorithm. The LSTM module has the tendency to handle the sequential and temporal dependencies effectively. This capacity of the LSTM helps the system to determine the order and timing of the network events, enabling the system to estimate the network traffic and predicting the attack patterns accurately. However, the traditional ML algorithms, such as SVM, RF, DNN, etc., often face challenges to capture long-range dependencies in sequences, leading to suboptimal performance. In addition, the memory cell and gating behavior enables the system to remember and forget data over long time intervals, allowing it to identify complex relations and patterns of network traffic over time intervals. These characteristics of LSTM make it unique from other ML approaches and aid the accurate classification of network traffics more than the other approaches. Moreover, the LSTM has the capacity to mitigate the exploding gradient problems, leading to more effective training under challenging data distributions. Furthermore, the architecture of LSTM enables it to learn and capture both contextual and hierarchical attributes. This structural property of LSTM enables it to differentiate normal and malicious activities more precisely than the other ML approaches. Thus, the usage of LSTM can be helpful to handle long-term intrusion issues. Moreover, when modelling complicated sequential IDS data, LSTMs are incredibly effective. The intensive assessment of the developed approach results proved that it has achieved better attack identification proficiency than than the earlier models, with performance metrices in terms of high recall, FPR, accuracy, F-measure, precision, and less FNR, MSE, and execution time.

Table 4. Overall comparative analysis for both the KDDCUP99 and UNSW-NB15 datasets.

KDDCUP99 Dataset					
Metrics and Methods	**MWKF-LSTM with DEDFA [26]**	**SA-IDPS [28]**	**BiLSTM [29]**	**GA-FCAM-CNN [32]**	**Proposed**
Accuracy	86	89	95.06	88	99.9
Recall	83	60	89	84	98.2
Precision	81	90	81.05	85	95
F-measure	82	59	84	82	98
FPR	0.001	0.045	0.005	0	0.09
FNR	1.34	0.91	0.65	1.85	0.001
UNSW-NB15 dataset					
Accuracy	86	89	95.06	88	99.9
Recall	83	60	89	84	98.2
Precision	81	90	81.05	85	95
F-measure	82	59	84	82	98
FPR	0.001	0.045	0.005	0	0.09
FNR	1.34	0.91	0.65	1.85	0.001

Previous attempts to design efficient security algorithms have failed. Experimental findings show that our suggested smart approach model, when used with important phases, effectively counters both common and uncommon assaults. In this research, the IDS performances are upgraded by incorporating deep LSTM networks with hybrid SSO feature selection methods by handling the uncertainties of data. The SSO algorithm can estimate the best and worst features from the database and obtained the optimal results. Because of this SSO algorithm, the best features are collected from the processed database and given to an LSTM algorithm for effective intrusion detection. Thus, this significant

detection is helpful for further effective intrusion prevention. The features in the proposed model are chosen by the SSO algorithm from the payload data and then utilized as input data by the LSTM layer. The unaltered unique payload situated at the initial phase of the flow is helpful in traffic categorization of unaltered data, even though the features chosen from the SSO algorithm contain significant data for the classification of traffic. As a result, it requires the least amount of preparation time and training. However, it also makes our approach well-suited for high-dimensional and large-scale domains. The execution time is reduced in this research due to the malicious traffic detection at the level of the packet. A thorough examination reveals that SHA3-256-based single-way hash functions can reliably identify and stop a variety of intrusion assaults on IoT networks, including spoofing, tampering, and replay attacks. The SHA3-256 was utilized in the proposed work, playing a significant role in mitigating the intrusions in the IoT by providing a strong cryptographic mechanism, thereby ensuring data integrity and confidentiality. In the developed approach, this hash function acts as an effective checksum mechanism, allowing the system to validate the integrity and confidentiality of the network traffic data. This cryptographic mechanism works by calculating the fixed-size unique hash value for the incoming network traffic data. This hash value acts as the digital fingerprint for the data and any modification in the data completely change its hash value. The proposed technique ensures data integrity by equating the generated hash value with the computed hash value; if the hash values remain unchanged, the data is secured. Otherwise, the system predicts it as injected data. Thus, it helps to detect any unauthorized alterations on the data. The integration of this hash function into the proposed approach enhances the security and promptly mitigate the attacks. In addition, the SHA3-256-based single-way hash function has fast processing speed, increased efficiency while handling huge volumes of data, and allows one-way security from attackers. Because hash algorithms are collision-resistant, it is improbable that two separate inputs would result in the same hash. Thus, the overall analysis consequences show that the developed model effectively detects and prevents various types of intrusion attacks, such as Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), Probing, Backdoors, Fuzzers Inquiry, Shellcode, Reconnaissance, and Worms, in IoT environments. The proposed approach accuracy value is 99.9%, which is more than the other methods for both KDDCUP99 and UNSW-NB15 databases. Our system has several benefits for quickening up the detection procedure, since it analyzes traffic at the packet level, such as the ability to disregard inspecting a significant number of packets in a transaction. The experimental findings demonstrate that, when compared to earlier work, our strategy is competitive and clearly superior regarding accuracy, recall, precision, FPR, detection percent, FNR, and F1-score. The algorithm that will be utilized to analyze and anticipate the incursion uses the level of accuracy as its primary performance metric. The major concern of the presented study is to addresses the security challenge associated with the IoT system. Typically, IoT systems contain a wide range of interconnected devices covering domains including smart homes, Industrial IoT (IIoT), healthcare units, agricultural sector, transportation, etc. Addressing the security issues in these domains is crucial for manifesting the proficiency of the designed algorithm. Combining the strengths of multiple algorithms, including LSTM, SSO, and SHA3-256, the developed approach offers a versatile, robust, and efficient mechanism for tackling the privacy and security problems in different domains. In all above-mentioned domains, the proposed technique follows the common protocol to mitigate the attacks, enabling secure data access and transmission. Moreover, the designed model performs better in a resource-constrained IoT environment. Typically, IoT devices are restricted to memory, power, and other energy resources. Therefore, an attack detection framework must consider these resource-constrained factors. The proposed model was designed with adaptability characteristics, such as the capacity for minimizing memory without losing its ability of capturing interconnections within the traffic data. In addition, the integration of SSO minimizes the computational complexity, making the feature selection process more sustainable and feasible. Moreover, the usage of SHA3-256 can mitigate the attacks without

producing a significant burden on hardware or power resources. Thus, the presented approach can be practicable for resource-constrained environments.

6. Conclusions

An artificial method for network intrusion detection and protection was created in this study. The analysis used the KDDCUP99 and UNSW-NB15 databases, which are available to the general public. Both datasets contain a substantial number of challenging columns. The model was designed using the LSTM technique, and the SSO was constructed for the best feature selection. Moreover, the SHA3-256 model was developed and the security of data transmission was improved. In comparison to the findings of the most recent publications in the literature, the model produced extremely good results. To estimate the robustness of the approach, the performance parameter accuracy was utilized. The model performed with 99.9% accuracy for both datasets. Also, the encryption and decryption times of the developed model showed rapid security function in the healthcare data transmission system. The work done on the dataset is outstanding, and the use of real-time network traffic includes the most advanced and sophisticated attack types. Future updates to the model are still possible. The suggested work's drawback is that it only tested network assaults online. It can also concentrate on creating an artificial method for detecting network threats worldwide in the future. In our upcoming work, we want to conduct more experiments by combining data from various disciplines and evaluating the usefulness of the suggested methodology. Also, in future, the multi-layer LSTM model can be designed for payload-assisted traffic classification in software-defined systems with specific databases.

Author Contributions: Conceptualization, A.M.A. and S.Q.; methodology, A.M.A., F.A., F.J.A. software, F.J.A.; validation, A.M.A. and S.Q.; formal analysis, F.A. investigation, F.A., F.J.A. resources, A.M.A.; data curation, S.Q.; writing–original draft preparation, S.Q.; writing–review and editing, F.A., F.J.A.; visualization, A.M.A. and S.Q.; supervision, A.M.A., F.A., project administration, A.M.A.; funding acquisition, A.M.A. All authors jointly worked on the results. All authors have read and agreed to the published version of the manuscript.

Funding: This research work was funded by Institutional Fund Projects under grant no. (IFPHI-091-611-2020) from the Ministry of Education and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

Data Availability Statement: The datasets that were used in this study are available online at the following links: https://www.kaggle.com/datasets/galaxyh/kdd-cup-1999-data, https://www.kaggle.com/datasets/dhoogla/unswnb15, accessed on 22 May 2023.

Acknowledgments: 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 IFPHI-091-611-2020 and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Nomenclature

IoT	Internet of Things
LSTM	Long Short-Term Memory
SSO	Shuffle Shepherd Optimization
IDS	Intrusion Detection System
SHA3-256	Secure Hash Algorithm 3 with a 256-bit hash value
DoS	Denial of Service
U2R	User to Root
R2L	Remote to Local
AI	Artificial Intelligence
ML	Machine Learning
CPU	Central Processing Unit
SVM	Support Vector Machine

KNN	K-nearest neighbor
RF	Random Forest
DNN	Deep neural network
CNN	Convolutional neural network
MACs	Message authentication codes
HE	Homomorphic Encryption
MWKF	Morlet Wavelet Kernel Function
DEDFA	Differential Evaluation-based Dragonfly Algorithm
E2CC	Enhanced Elliptical Curve Cryptography
ANN	Artificial neural network
CH	Cluster Heads
HHO	Harris Hawks Optimization
CSO	Chicken Swarm Optimization
DBN	Deep Belief Network
CAVO	Chronological Anticorona Virus Optimization
IFS	Interplanetary File System
FCM	Fuzzy C-means clustering
GA	Genetic algorithm
BiLSTM	Bidirectional Long Short-Term Memory
MSE	Mean Square Error
DPI	Deep Packet Inspection
RAM	Random Access Memory
FNR	False Negative Rate
FPR	False Positive Rate
$x_{m,n}^{norm}$	Min-max normalization
n	Feature count
$x_{m,n}$	Variable assessment
y_m	Feature n of $x_{m,n}$
j	Quantity of training or testing set
$\min(x_{:,n})$	Minimum rate of the column feature
$\max(x_{:,n})$	Maximum rate of the column feature
$L_{\min}$	Lowest archetypal parameter limits
$L_{\max}$	Supreme archetypal parameter limits
$rand$	Random number
g	Number of individuals in each collection
h	Entire number of collections
u	Index of the community
v	Index of the members within the community
L_p	Multi-community matrix
$K_{u,v}^{w}$	Possibility of looking into additional regions of the solution space
$K_{u,v}^{b}$	Ability to look around previously visited potential solution space regions
$K_{u,v}$	Step size
$L_{u,v}$	Objective function values
$rand_1$ and $rand_2$	Random parameters
$L_{u,b}$	Objective solution with the best parameter
$L_{u,w}$	Objective solution with worst parameter
χ	Variable influencing explorations
ζ	Variable influencing exploitation
x_t	Input gate
p_t	Output gate
W_p	Hidden state weight
b_p	Concealed layer has a bias
tanh	Sigmoid layer function
E_t	Potential parameter
$\hat{E}_t$	New parameter
W	Weight matrices
b	Bias matrix
q_t	Output values

y_t	Output cell state
W_y	Output gate's weight
b_y	Bias matrix
f	Permutation function
S	Rate
D	Capacity
y	Bit string
x	Input bit string
P	Padding task
k	Bit blocks size
m	Output length
R	Padded bit string

References

1. Park, S.; Choi, G.J.; Ko, H. Information technology-based tracing strategy in response to COVID-19 in South Korea—Privacy controversies. *JAMA* **2020**, *323*, 2129–2130. [CrossRef]
2. Miśkiewicz, R. The impact of innovation and information technology on greenhouse gas emissions: A case of the Visegrád countries. *J. Risk Financ. Manag.* **2021**, *14*, 59. [CrossRef]
3. Gopal, G.; Suter-Crazzolara, C.; Toldo, L.; Eberhardt, W. Digital transformation in healthcare-architectures of present and future information technologies. *Clin. Chem. Lab. Med. (CCLM)* **2021**, *57*, 328–335. [CrossRef]
4. Mansour, R.F.; El Amraoui, A.; Nouaouri, I.; Diaz, V.G.; Gupta, D.; Kumar, S. Artificial intelligence and internet of things enabled disease diagnosis model for smart healthcare systems. *IEEE Access* **2021**, *9*, 45137–45146. [CrossRef]
5. Chawla, N. AI, IOT and wearable technology for smart healthcare? A review. *Int. J. Green Energy* **2020**, *7*, 9–13.
6. Kumar, S.; Raut, R.D.; Narkhede, B.E. A proposed collaborative framework by using artificial intelligence-internet of things (AI-IoT) in COVID-19 pandemic situation for healthcare workers. *Int. J. Healthc. Manag.* **2020**, *13*, 337–345. [CrossRef]
7. Javaid, M.; Khan, I.H. Internet of Things (IoT) enabled healthcare helps to take the challenges of COVID-19 Pandemic. *J. Oral Biol. Craniofacial Res.* **2021**, *11*, 209–214. [CrossRef] [PubMed]
8. Chaudhary, V.; Kaushik, A.; Furukawa, H.; Khosla, A. Review—Towards 5th Generation AI and IoT Driven Sustainable Intelligent Sensors Based on 2D MXenes and Borophene. *ECS Sens. Plus* **2022**, *1*, 013601. [CrossRef]
9. Elvira, N.; Stehel, V. Internet of things sensing networks, artificial intelligence-based decision-making algorithms, and real-time process monitoring in sustainable industry 4.0. *J. Self-Gov. Manag. Econ.* **2021**, *9*, 35–47.
10. Popa, A.; Hnatiuc, M.; Paun, M.; Geman, O.; Hemanth, D.J.; Dorcea, D.; Son, L.H.; Ghita, S. An intelligent IoT-base food quality monitoring approach using low-cost sensors. *Symmetry* **2019**, *11*, 374. [CrossRef]
11. Greco, L.; Percannella, G.; Ritrovato, P.; Tortorella, F.; Vento, M. Trends in IoT based solutions for health care: Moving AI to the edge. *Pattern Recognit. Lett.* **2020**, *135*, 346–353. [CrossRef] [PubMed]
12. Jain, A.; Singh, T.; Sharma, S.K. Security as a solution: An intrusion detection system using a neural network for IoT enabled healthcare ecosystem. *Interdiscip. J. Inf. Knowl. Manag.* **2021**, *16*, 331–369. [CrossRef] [PubMed]
13. Eskandari, M.; Janjua, Z.H.; Vecchio, M.; Antonelli, F. Passban IDS: An intelligent anomaly-based intrusion detection system for IoT edge devices. *IEEE Internet Things J.* **2020**, *7*, 6882–6897. [CrossRef]
14. Arisdakessian, S.; Wahab, O.A.; Mourad, A.; Otrok, H.; Guizani, M. A survey on iot intrusion detection: Federated learning, game theory, social psychology and explainable ai as future directions. *IEEE Internet Things J.* **2022**, *10*, 4059–4092. [CrossRef]
15. Öztürk, T.; Turgut, Z.; Akgün, G.; Köse, C. Machine learning-based intrusion detection for SCADA systems in healthcare. *Netw. Model. Anal. Health Inform. Bioinform.* **2022**, *11*, 47. [CrossRef]
16. Chen, S.; Webb, G.I.; Liu, L.; Ma, X. A novel selective naïve Bayes algorithm. *Knowl.-Based Syst.* **2020**, *192*, 105361. [CrossRef]
17. Ponmalar, A.; Dhanakoti, V. An intrusion detection approach using ensemble support vector machine based chaos game optimization algorithm in big data platform. *Appl. Soft Comput.* **2022**, *116*, 108295. [CrossRef]
18. Balyan, A.K.; Ahuja, S.; Lilhore, U.K.; Sharma, S.K.; Manoharan, P.; Algarni, A.D.; Elmannai, H.; Raahemifar, K. A hybrid intrusion detection model using ega-pso and improved random forest method. *Sensors* **2022**, *22*, 5986. [CrossRef]
19. Wahab, O.A. Intrusion detection in the iot under data and concept drifts: Online deep learning approach. *IEEE Internet Things J.* **2022**, *9*, 19706–19716. [CrossRef]
20. Mohammed, R.A.; Ali Alheeti, K.M. Intrusion detection system for Healthcare based on Convolutional Neural Networks. In Proceedings of the 2022 Iraqi International Conference on Communication and Information Technologies (IICCIT), Basrah, Iraq, 7–8 September 2022.
21. Wahab, F.; Zhao, Y.; Javeed, D.; Al-Adhaileh, M.H.; Almaaytah, S.A.; Khan, W.; Saeed, M.S.; Shah, R.K. An AI-driven hybrid framework for intrusion detection in IoT-enabled E-health. *Comput. Intell. Neurosci.* **2022**, *2022*, 6096289. [CrossRef]
22. Jayalaxmi, P.; Saha, R.; Kumar, G.; Conti, M.; Kim, T.H. Machine and Deep Learning Solutions for Intrusion Detection and Prevention in IoTs: A Survey. *IEEE Access* **2022**, *10*, 121173–121192. [CrossRef]
23. Alsoufi, M.A.; Razak, S.; Siraj, M.M.; Nafea, I.; Ghaleb, F.A.; Saeed, F.; Nasser, M. Anomaly-based intrusion detection systems in iot using deep learning: A systematic literature review. *Appl. Sci.* **2021**, *11*, 8383. [CrossRef]

24. Kumaar, M.A.; Samiayya, D.; Vincent, P.M.D.R.; Srinivasan, K.; Chang, C.-Y.; Ganesh, H. A Hybrid Framework for Intrusion Detection in Healthcare Systems Using Deep Learning. *Front. Public Health* **2022**, *9*, 2295.
25. Asharf, J.; Moustafa, N.; Khurshid, H.; Debie, E.; Haider, W.; Wahab, A. A review of intrusion detection systems using machine and deep learning in internet of things: Challenges, solutions and future directions. *Electronics* **2022**, *9*, 1177. [CrossRef]
26. Nave, O. Modification of semi-analytical method applied system of ODE. *Mod. Appl. Sci.* **2020**, *14*, 75. [CrossRef]
27. Kumar, K.P.; Retnaswamy, B. A Novel MWKF-LSTM Based Intrusion Detection System for the IoT-Cloud Platform with Efficient User Authentication and Data Encryption Models. *Res. Sq.* **2022**, preprint. [CrossRef]
28. Pooja, T.S.; Shrinivasacharya, P. Evaluating neural networks using Bi-Directional LSTM for network IDS (intrusion detection systems) in cyber security. *Glob. Transit. Proc.* **2022**, *2*, 448–454.
29. Islabudeen, M.; Devi, M.K.K. A smart approach for intrusion detection and prevention system in mobile ad hoc networks against security attacks. *Wirel. Pers. Commun.* **2022**, *112*, 193–224. [CrossRef]
30. Alkadi, O.; Moustafa, N.; Turnbull, B.; Choo, K.K.R. A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks. *IEEE Internet Things J.* **2022**, *8*, 9463–9472. [CrossRef]
31. Mansour, R.F. Blockchain assisted clustering with Intrusion Detection System for Industrial Internet of Things environment. *Expert Syst. Appl.* **2022**, *207*, 117995. [CrossRef]
32. Saviour, M.P.A.; Samiappan, D. IPFS based storage Authentication and access control model with optimization enabled deep learning for intrusion detection. *Adv. Eng. Softw.* **2023**, *176*, 103369. [CrossRef]
33. Nguyen, M.T.; Kim, K. Genetic convolutional neural network for intrusion detection systems. *Future Gener. Comput. Syst.* **2022**, *113*, 418–427. [CrossRef]
34. Almalawi, A.; Khan, A.I.; Alsolami, F.; Abushark, Y.B.; Alfakeeh, A.S. Managing Security of Healthcare Data for a Modern Healthcare System. *Sensors* **2023**, *23*, 3612. [CrossRef] [PubMed]
35. Laghrissi, F.; Douzi, S.; Douzi, K.; Hssina, B. Intrusion detection systems using long short-term memory (LSTM). *J. Big Data* **2021**, *8*, 65. [CrossRef]
36. Dao, P.N.; Liu, Y. Adaptive reinforcement learning in control design for cooperating manipulator systems. *Asian J. Control* **2022**, *24*, 1088–1103. [CrossRef]
37. Chen, L.; Dai, S.-L.; Dong, C. Adaptive Optimal Tracking Control of an Underactuated Surface Vessel Using Actor-Critic Reinforcement Learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, 1–14. [CrossRef] [PubMed]
38. Fang, M.; Boutros, F.; Damer, N. Unsupervised face morphing attack detection via self-paced anomaly detection. In Proceedings of the 2022 IEEE International Joint Conference on Biometrics (IJCB), Abu Dhabi, United Arab Emirates, 10–13 October 2022.
39. Aouedi, O.; Piamrat, K.; Muller, G.; Singh, K. Federated semisupervised learning for attack detection in industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2022**, *19*, 286–295. [CrossRef]