IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 20 February 2024, accepted 27 April 2024, date of publication 7 May 2024, date of current version 15 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3397303

RESEARCH ARTICLE

Hybrid Chaotic Zebra Optimization Algorithm and Long Short-Term Memory for Cyber Threats Detection

REHAM AMIN[1], GHADA EL-TAWEEL[1], AHMED F. ALI[1,2], AND MOHAMED TAHOUN[1]

[1]Faculty of Computers and Informatics, Suez Canal University, Ismailia 8366004, Egypt
[2]Faculty of Information Technology and Computer Science, Sinai University, Kantara 8392130, Egypt

Corresponding author: Reham Amin (reham_amin@ci.suez.edu.eg)

ABSTRACT Cyber Threat Detection (CTD) is subject to complicated and rapidly accelerating developments. Poor accuracy, high learning complexity, limited scalability, and a high false positive rate are problems that CTD encounters. Deep Learning defense mechanisms aim to build effective models for threat detection and protection allowing them to adapt to the complex and ever-accelerating changes in the field of CTD. Furthermore, swarm intelligence algorithms have been developed to tackle the optimization challenges. In this paper, a Chaotic Zebra Optimization Long-Short Term Memory (CZOLSTM) algorithm is proposed. The proposed algorithm is a hybrid between Chaotic Zebra Optimization Algorithm (CZOA) for feature selection and LSTM for cyber threat classification in the CSE-CIC-IDS2018 dataset. Invoking the chaotic map in CZOLSTM can improve the diversity of the search and avoid trapping in a local minimum. In evaluating the effectiveness of the newly proposed CZOLSTM, binary and multi-class classifications are considered. The acquired outcomes demonstrate the efficiency of implemented improvements across many other algorithms. When comparing the performance of the proposed CZOLSTM for cyber threat detection, it outperforms six innovative deep learning algorithms for binary classification and five of them for multi-class classification. Other evaluation criteria such as accuracy, recall, F1 score, and precision have been also used for comparison. The results showed that the best accuracy was achieved using the proposed algorithm for binary is 99.83%, with F1-score of 99.82%, precision of 99.83%, and recall of 99.82%. The proposed CZOLSTM algorithm also achieved the best performance for multi-class classification among other compared algorithms.

INDEX TERMS Cyber security, deep learning, feature selection, long short-term memory, intrusion detection, swarm intelligence, zebra optimization algorithm.

I. INTRODUCTION

The communication revolution is regarded as being crucial to day-to-day life since it serves as the foundation for emerging smart devices. This change presents major challenges even though it creates new opportunities. Being able to adjust to the complicated and ever-accelerating changes in the field of Cyber Threat Detection (CTD) is the most significant challenge. Poor accuracy, high learning complexity, limited scalability, and a high false positive rate (FPR) are just a few of the problems that CTD faces [1]. The distributed nature of the networking environment makes it susceptible to intrusion attacks. Therefore, CTD offers a variety of network security technologies as a result. Firewalls, Intrusion Prevention Systems (IPSs), and Intrusion Detection Systems (IDSs) are some of the most popular network security solutions [2].

The primary goal of Deep learning (DL) algorithms for the field of CTD is to classify network data which helps to improve the process of threat detection and defense. Denial-of-service (DoS) attacks, data spying, spoofing, and network resource occupancy are just a few examples of security issues that are constantly changing that the DL-based algorithm

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

can handle when it comes to CTD [3], [4]. Deep learning algorithms operate on high-dimensional datasets [5], which are significantly impacted by extensive model development processes, redundant data, and decreased performance, all of which make data analysis extremely challenging [6], [7]. The most often used DL algorithms are Deep Belief Network (DBN), Recurrent Neutral Network (RNN), Convolutional Neutral Network (CNN), Long Short Term Memory (LSTM), and etc.

Feature selection (FS) is used as a significant preprocessing step to address this problem. Its objectives are to select the optimal features by eliminating noisy, superfluous, and confusing data. According to [8], there are two main steps of FS algorithms which are: search strategy and subset quality evaluation. The search strategy selects subsets of features. In the later step, the quality of the selected subsets that the search strategy module produced is assessed using a classifier. FS algorithms are widely divided into filter-based algorithms and wrapper-based algorithms.

The features are rated using filter-based algorithms, then the highest-scored features are selected and fed into a predictor. In contrast, the wrapper-based algorithms, which is the benchmark for feature selection, use a search algorithm that will identify the subset that offers the best predictor performance [9]. Statistical data dependencies are used in the filter-based algorithms. whereas machine learning algorithms are used in wrapper-based algorithms.

Although filter-based algorithms are relatively rapid, they cannot process redundant data. These algorithms employ measurements like the distance between the dimensions, correlation between the dimensions, consistency, and so on. Their methodologies include: principal component analysis [10], mutual information [11], and information gain [12]. However, the wrapper-based algorithms outperform the filter-based ones due to the employment of classifiers in the evaluation module [13].

In recent years, natural-inspired metaheuristic algorithms have gained prominence due to their strong and effective ability to handle complex real-world problems [14]. Among the issues that optimization algorithms resolve are cognizable crime rate prediction, cancer classification, and gene selection optimization [15], [16]. These algorithms can identify the best solutions taking advantage of the useful information of the population [8]. The nature-inspired algorithms have been developed to tackle challenging optimization problems. These algorithms are generally divided into swarm intelligence algorithms and evolutionary algorithms [17]. The two main categories of metaheuristic algorithms are the single solution and population solution-based algorithms [14]. Both categories differ in the way used to locate a solution. For a single solution, all possible solutions are created at random before the ideal one is found. While, the population-based algorithm generates a random number of solutions, and iteratively updates the values of each solution. Iterative development is used to create the optimal solution [18]. According to [19], the most well-known algorithms for roughly solving the hardest problems available today are: Artificial Bee Colony (ABC) [20], Monarch Butterfly Optimization (MBO) [21], Bat Algorithm (BA) [22], Cuckoo Search (CS) [23], Differential Evolution (DE) [24], Firefly Algorithm (FA) [25], Particle Swarm Optimization (PSO) [26], etc.

The Artificial Bee Colony (ABC) [20] draws inspiration from the intelligent food-seeking tactics employed by bee colonies. To obtain a substantial amount of food supply, a honey bee colony can be concurrently expanded in several directions over a distance of 10 to 14 kilometers. It has been widely applied to optimization challenges in fields such as computer image processing and manufacturing. The Monarch Butterfly Optimization (MBO) [21] simulates the migration routes traveled by American monarch butterflies. It has been applied to classification, data clustering, optimization issues, and other tasks. Bat Algorithm (BA) [22] is a pulse emissivity and loudness-based echolocation algorithm inspired by the behavior of bats. It is used in fuzzy logic, image processing, data mining, clustering, classification, and other applications. Inspired by the breeding practices of cuckoo species and the flying patterns of some birds and fruit flies, Cuckoo Search (CS) [23] was developed. It never makes a nest of its own to deposit eggs. Rather, it lays its eggs in the nests of other birds. Global optimization, job scheduling, and speech recognition are among the applications using this algorithm.

Differential Evolution (DE) [24] takes inspiration from the differential information of several individuals to represent the evolutionary individual conflict based on group differences. Feature selection, network intrusion detection, classification, and other applications have all made use of it. Firefly Algorithm (FA) [25] is an algorithm that mimics the fluttering and flashing behaviors of fireflies. To find food and mates, and to communicate with one another, fireflies flash signals to one another. It has been used for feature selection and optimization issues. The feeding patterns of bird flocks serve as an inspiration for Particle Swarm Optimization (PSO) [26]. Multi-objective problem optimization, classification, pattern recognition, biological system modeling, prediction and forecasting, networking systems, scheduling, robotics, fuzzy systems, security and military applications, finance, and other fields are among the many uses of PSO.

There is no guarantee that an optimization algorithm will work efficiently for every optimization problem due to the randomization-based feature selection algorithms [8]. This could lead to decreased performance when trying to address certain problems. This issue motivated us to conduct an evaluation of the recently proposed Zebra optimization algorithm (ZOA). The ZOA algorithm is a novel meta-heuristic that is inspired from the foraging habits and predator defense mechanisms of zebras. Because ZOA outperforms other optimization algorithms in diverse applications and has the potential to be an optimization technique for global optimization issues, it has drawn to be the interest of numerous academics. To some degree, local

optima stagnation can be prevented by using this population-based metaheuristic. It is also capable of convergently approaching the optimal value. To the best of our knowledge, the feature selection problem has never been systematically solved using ZOA. This is the primary factor that motivated us to use ZOA as the base of our research.

Swarm intelligence optimization algorithms rely on pseudo-random numbers, even though they would increase efficiency and decrease time complexity. Chaotic maps were frequently used in place of the pseudo-random numbers that were distributed in the Gauss distribution in computer engineering. The chaotic optimization method was named after these optimization advancements [27]. To improve chaotic qualities for a wide range of system parameters, a new chaotic system must be developed because chaotic maps are computationally expensive and difficult to apply [28].

In this paper, a Chaotic variant of the Zebra Optimization Algorithm (CZOLSTM) based on a hybrid Hénon and Gingerbread Chaotic Maps is proposed. The proposed algorithm invokes chaotic maps in ZOA to increase the diversity of the search process and to avoid trapping in the local minimum. The main purpose of the suggested CZOLSTM algorithm is to detect cyber threats, particularly intrusions, and to minimize or stop illegal access to any system, independent of the system's discipline. Any system need essentially be protected from unauthorized attacks, which is the primary goal of the proposed CZOLSTM algorithm. However, this does not imply that it cannot handle other technological challenges. Future work will examine the algorithm's effectiveness in a variety of contexts by taking multidisciplinary case studies into account and assessing CZOLSTM's performance using multipurpose datasets. Below is the main contribution of the paper:

- We have proposed a hybrid algorithm called CZOLSTM that combines the LSTM algorithm for cyber threat detection with the Chaotic Zebra optimization Algorithm (CZOA) for feature selection to reduce the dimensionality of the CSE-CIC-IDS2018 dataset. To the best of our knowledge, this is the first time that the CZOA has been used for this purpose.
- The optimal 39 features out of the 79 features were selected using the CZOA. Additionally, we evaluated the efficiency of the proposed CZOLSTM using the CSE-CIC-IDS2018 dataset aginst the standard LSTM in terms of the number of selected features based on the outcomes of each classification mode (binary or multiclass).
- In terms of performance comparison, the proposed CZOLSTM for cyber threat identification works better than five of the most recent deep learning algorithms for multiclass classification and six of them for binary classification.

The rest of this paper is organized as follows: the related work of the most popular and current DL algorithm applied to the CSE-CIC-IDS2018 dataset is provided in Section II. Section III provides a detailed explanation of our proposed algorithm. Section IV explains the experimental setup, parameter adjustment, the obtained results, and a comparison with other algorithms. Finally, the paper is concluded in Section V, where the future perspectives are presented.

II. LITERATURE REVIEW

Amin et al. [29] proposed a Hybrid XG Boosted and Long Short-Term Memory algorithm (HXGBLSTM). Six well-known evolutionary computation algorithms were compared to the newly created Zebra Optimization Algorithm (ZOA). These algorithms include the Genetic Algorithm as an Evolutionary Algorithm, the Particle Swarm Optimization Algorithm, the Bio-inspired Algorithms, the Bat Optimization Algorithm, the Firefly Optimization Algorithm, and the Monarch Butterfly Optimization Algorithm. The outcomes were compared with the wrapper-based XGBoost feature selection algorithm using the CSE-CIC-IDS2018 dataset. They obtained results of 99.8% for binary classification and around 100% for multi-class classification.

Alzughaibi and El Khediri [30] proposed two Deep Neural Network (DNN) models for the CSE-CIC-IDS2018 dataset: one based on a Multi-Layer Perceptron (MLP) with Back Propagation (BP), the other one is an MLP with Particle Swarm Optimization (PSO). The obtained results were 98.41% for multi-class classification and 98.97% for binary classification. The model's performance was respectable, but as it was only tested on a single dataset, its true applicability might not have been demonstrated. Additionally, the study did not take into account training and validation times, which may affect how usable the results are for future studies. Furthermore, in the multiclass classification scenario, the obtained result is not compared with other relevant work.

Liu et al. [31] suggested using feature engineering and machine learning in Software-Defined Networks (SDN) to detect Distributed Denial of Service (DDoS) attacks. They reduced the number of features in the dataset from 79 to 26 by using an enhanced binary grey wolf optimization approach for feature extraction. Then, they assessed and chose the best classifier for the original and feature-extracted datasets, respectively, using five machine learning models: Random Forest (RF), Support Vector Machine (SVM), ZOA, Decision Tree, and K Nearest Neighbour (kNN). According to the results, the RF classifier had the best performance in terms of F1 score, accuracy, precision, recall, and 0.9913, 0.9843, 0.9992, and 0.9913, respectively. Nevertheless, the authors did not contrast machine learning outcomes with those of other deep learning methods. Also, multiclass classification is not mentioned.

To increase security in IDS, Donkol et al. [32] presented the Enhanced Long-Short Term Memory with Recurrent Neural Network (RNN) (ELSTM-RNN) algorithm. The gradient-clipping problem is solved by the proposed system using enhanced LSTM classification and probable point particle swarm optimization (LPPSO). For further validation, the proposed approach has been implemented on the UNSW-NB15, CICIDS2017, CSE-CIC-IDS2018, and BOT datasets. Based on the findings, the proposed ELSTM-RNN

framework is assessed on several metrics, such as accuracy, precision, recall, and error rate. With respect to accuracy, the proposed technique performed better than 98% for the CSE-CIC-IDS2018 dataset. However, the obtained performance indicators were not mentioned numerically by the authors.

Babu and Rao [33] proposed using Improved Monarchy Butterfly optimization Algorithm (IMBO) Based optimized ANU-Net for Intrusion Detection. For testing, datasets like NSLKDD, CIC-IDS 2017, and CSE-CIC-IDS 2018 were used. The proposed model achieved 98.87% accuracy when compared to other algorithms. However, multiclass classification was not mentioned.

Ghanbarzadeh et al [34] presented a novel approach to intrusion detection in computer networks called MQBHOA, or the multi-objective, balanced, and quantum-inspired Horse herd optimization algorithm. The performance of the novel algorithm was assessed using two distinct data sets, NSL-KDD and CSE-CIC-IDS2018. It was contrasted with a number of other algorithms in feature selection and classification, including Grey Wolf Optimizer (GWO), Whale Optimization Algorithm (WOA), FireFly Algorithm (FFA), Moth-Flame Optimization (MFO), Particle Swarm Optimization (PSO), KNN, RF, Naïve Bayes (NB), and Self Organisation Map (SOM). It achieved a precision of 99.56 and an accuracy of 99.78. Nevertheless, the study did not take multiclass classification into account.

Fraihat et al. [35] extracted the optimal features by combining machine learning (ML) with an altered version of the Arithmetic Optimization Algorithm (AOA). A pair of machine learning models, Random Forest and Extra Trees, are trained with the selected features. IoT traffic datasets including (NF-CSE-CIC-IDS2018-v2, NF-ToN-IoT-v2, and NF-UNSW-NB15-v2) which are built up of a shared collection of 43 NetFlow-based features were used in analysis. The proposed AOA achieved an accuracy of 99.54% and 99.52%, for binary and multi-classification respectively.

Bhardwaj and Dave [36] proposed Forensics Exploratory Data Analysis (FEDA) as a general tool for attack investigation combined with Convolution Neural Networks (CNN). Datasets from UNSW-NB15, CSE-CIC-IDS2018 (CIC2018), and CIC-Darknet2020 are utilized to evaluate the performance of the suggested system. The model achieved accuracy of 98.3% and 99.4%, for binary and multi-class classification, respectively.

To improve learning from imbalanced samples, Zhang and Liu [37] presented IoT intrusion detection based on data augmentation. The authors used ICVAE-BSM, which stands for Improved Conditional Variational Autoencoder and Borderline Synthetic Minority Oversampling Technique. The system obtained an accuracy of 98.67, an F1 score of 98.50, a recall of 98.95, and a precision of 98.04.

Black Widow Optimized Convolutional Long Short-Term Memory (BWO-CONV-LSTM) neural networks on a MapReduce-based platform were presented by Kanna and Santhi [38]. Artificial Bee Colony (ABC) algorithm is used to extract the optimal features. The proposed system is combining convolutional and LSTM neural networks. The suggested BWO-CONV-LSTM network optimizes hyper-parameters to produce the optimal design. The NSL-KDD, ISCX-IDS, UNSW-NB15, and CSE-CIC-IDS2018 datasets are used to assess the performance of the BWO-CONV-LSTM based IDS model. The results show that the proposed BWO-CONV-LSTM model has high intrusion detection performance with 98.67%, 97.003%, 98.667%, and 98.25% accuracy, for the NSL-KDD, ISCX-IDS, UNSW-NB15, and CSE-CIC-IDS2018 datasets, respectively. The performance was obtained with fewer false values, less computation time, and better classification coefficients. Tables 1 and 2 represent a comparison among the studies in the reviewed literature. The related literature is compared with respect to the following: the description of the algorithm, the dataset that was used, the type of detection problem (using either machine learning (ML) or deep learning (DL)), the tested classification mode (binary or multi-class), the results obtained, and the drawbacks.

III. THE PROPOSED CZOLSTM ALGORITHM

The details of the proposed CZOLSTM algorithm are presented in this section. The primary algorithm, the dataset's preparation and preprocessing, and evaluation metrics are all provided. Furthermore, the time complexity of the proposed CZOLSTM is discussed in Section III-D3. All of the symbols used throughout the paper are listed in Table 3.

A. ZEBRA OPTIMIZATION ALGORITHM

The zebra's natural behavior serves as the primary source of inspiration for the Zebra Optimization Algorithm (ZOA), a novel swarm intelligence algorithm [39]. Zebra's foraging behavior and its defense mechanism against predator attacks are simulated by ZOA. The most significant social behaviors exhibited by zebras in the wild are foraging and defense strategy against predators [39], [40].

In their foraging behavior, zebras consume mostly grass and plant materials like leaves and sprouts because they are herbivores. Zebras move to forage when a pioneer zebra leads the way. Thus, this pioneer zebra leads the other zebras in the herd as they go across the plains [41].

While in defense behavior against predators, the zebra is a gregarious animal that always resides in a group to protect itself from predators [42]. The animal's natural behavior is to flee from the predator in a zigzag pattern, but occasionally it may group together to create a defensive structure, which will confuse or frighten the predator.

The simulation findings demonstrate that ZOA outperforms other optimization algorithms by striking an appropriate balance between exploration and exploitation [39]. After providing a description, the ZOA steps are modeled mathematically in the next few lines.

The population of zebras can be modeled by the $N \times m$ matrix X which is specified in Equation 1

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N \times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,m} \end{bmatrix} \quad (1)$$

where X is the zebra population, X_i is the ith zebra in the search space, $x_{i,j}$ is the value of the jth problem variable proposed by the ith zebra, N is the number of zebras, and m is the number of decision variables.

Every zebra symbolizes a potential solution to the optimization problem. Consequently, the suggested values of each zebra for the problem variables can be used to evaluate the objective function. The objective function can be expressed as a vector using Equation 2

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N \times 1} \quad (2)$$

The next few lines consider in detail the two natural behaviors of zebras which include:

- Phase 1: Foraging Behavior
- Phase 2: Defense strategies against predators

a: PHASE 1: FORAGING BEHAVIOR

One particular type of zebra is known as the plain zebra. It is considered a pioneer grazer. It feeds on the higher, less nutritional grass canopy, creating an environment favorable for other species that require the shorter, more nutritious grasses to be below [41]. Regarding ZOA, the best individual within the population is called the pioneer zebra, guiding the rest of the population toward its location in the search space. Consequently, Equations 3 and 4 can describe how zebras adjust their position in the search space throughout the foraging phase.

$$X_i^t = \begin{cases} X_i^{t+1}, & \text{if } F_i^{t+1} < F_i^t. \\ X_i^t, & \text{otherwise.} \end{cases} \quad (3)$$

$$x_{i,j}^{t+1} = x_{i,j}^t + r.(PZ_j - I.x_{i,j}^t) \quad (4)$$

$$I = round(1 + rand) \quad (5)$$

where X_i^{t+1} is the new status of the ith zebra based on the first phase $P1$, t is the current iteration, $x_{i,j}^{t+1}$ is the j^{th} dimension value for the i^{th} zebra, r is random number in the interval $[0, 1]$, PZ_j is the jth dimension of the Pioneer Zebra (Best Solution), $rand$ is random number in the interval $[0, 1]$, and $I \in 1, 2$ if $I = 2$ this means more changes in the population movement.

b: PHASE 2: DEFENSE STRATEGIES AGAINST PREDATORS

Zebras must decide between two strategies when threatened by other predators: flee or defend [42]. In the first strategy (S1), the zebra chooses an escape route in response to a lion's attack. In contrast, if a smaller predator (such as a dog) attacks, the zebra would use the aggressive strategy (S2) to scare it away. As a result, zebras' movements in the search space during the defense phase can be explained by Equations 6 and 7.

$$X_i^t = \begin{cases} X_i^{t+1}, & \text{if } F_i^{t+1} < F_i^t. \\ X_i^t, & \text{otherwise.} \end{cases} \quad (6)$$

where X_i^{t+1} is the new status of the ith zebra based on the second phase $P2$, R is a constant number $= 0.01$, r is a random number in the interval $[0, 1]$, t is the iteration contour, T_{max} is the maximum number of iterations, P_s is the probability of choosing one of the two strategies that are randomly generated in the interval $[0, 1]$, AZ_j is the jth dimension of the Attacked Zebra. The Zebra Optimization Algorithm is shortened to Algorithm 1.

$$x_{i,j}^{t+1} = \begin{cases} S1 : x_{i,j}^t + R.(2r - 1).(1 - \dfrac{t}{T_{max}}).x_{i,j}^t, \\ \quad \text{if } P_s <= 0.5. \\ S2 : x_{i,j}^t + r.(AZ_j - I.x_{i,j}^t), \\ \quad \text{otherwise.} \end{cases} \quad (7)$$

where X_i^{t+1} is the new status of the ith zebra based on the second phase $P2$, R is a constant number $= 0.01$, r is a random number in the interval $[0, 1]$, t is the iteration contour, T_{max} is the maximum number of iterations, P_s is the probability of choosing one of the two strategies that are randomly generated in the interval $[0, 1]$, AZ_j is the jth dimension of the Attacked Zebra. The Zebra Optimization Algorithm is shortened to Algorithm 1.

B. CHAOTIC MAPS

The dynamic nature of chaotic maps makes it easier for optimization algorithms to scan the search space more deeply and dynamically. To generate a chaotic sequence, there are several mathematical operations known as chaotic maps that could be used [27]. Random sequences could be substituted by any number of chaotic maps [43]. For example, the Gingerbreadman chaotic map is an outstanding example of chaotic randomness in two dimensions [44]. The two dimensions of the Gingerbreadman chaotic map are displayed using Equation 8. It was selected due to its superior performance over other chaotic maps in previous research [45].

$$\begin{aligned} x_{n+1} &= 1 - y_n + |x_n|, \\ y_{n+1} &= x_n \end{aligned} \quad (8)$$

The two dimensions of the Gingerbreadman chaotic map must be reduced to just one in order to satisfy our purpose.

TABLE 1. Comparison among the studies in the reviewed literature.

Research Study	Algorithm Description	Dataset	Detection problem (Binary / Muti-class)	Classification Mode (ML / DL)	Results	Cons
[29]	The proposed algorithm used a Hybrid XG Boosted and Long Short-Term Memory (HXG-BLSTM). The selected optimal features were 21. Six well-known evolutionary computation algorithms were compared to the newly created Zebra Optimization Algorithm (ZOA). These algorithms include the Genetic Algorithm as an Evolutionary Algorithm, the Particle Swarm Optimization Algorithm, the Bio-inspired Algorithms, the Bat Optimization Algorithm, the Firefly Optimization Algorithm, and the Monarch Butterfly Optimization Algorithm	CSE-CIC-IDS2018	Both	DL	99.8% for binary and 100% for multi-class classification	The authors have not considered the time complexity of the proposed algorithm
[30]	The proposed model used a Multi-Layer Perceptron with Back Propagation for classification. For feature selection purpose, authors utilized Particle Swarm Optimization. The selected optimal features were 24. The model's performance was respectable, and the system was evaluated for both binary and multiclass classification	CSE-CIC-IDS2018	Both	DL	98.97% for binary and 98.41% for multiclass classifications	PSO achieved less performance than expected, It required a great deal of training time and hardware resources, and , in the multiclass classification scenario, the obtained result is not compared with the other relevant work.
[31]	Five machine learning models (e.g.: RF, SVM, ZOA, Decision Tree, and kNN) were used for classification and an enhanced binary grey wolf optimization approach was used for feature extraction. 26 features were chosen as the ideal features.	CSE-CIC-IDS2019	Both	Both	98.43% for binary classification	The authors did not contrast machine learning outcomes with those of other deep learning methods. Also, the multiclass classification is not mentioned.
[32]	For classification, the proposed algorithm employed Enhanced Long-Short Term Memory with Recurrent Neural Network (ELSTM-RNN). The authors used probable point particle swarm optimization for the purpose of feature selection. 38 features were chosen as the ideal features.The gradient-clipping problem is solved by the proposed algorithm.	UNSW-NB15, CIC-IDS-2017, CSE-CIC-IDS2018, and BOT	Binary	DL	98% for binary classification	The obtained performance indicators were not mentioned numerically by the authors.
[33]	An optimized ANU-Net was employed in the proposed model for for intrusion detection. Improved Monarchy Butterfly Optimization Algorithm was used by the authors for feature selection purposes. Multiple datasets were tested based on the propsed algorithm.	NSLKDD, CIC-IDS 2017, and CSE-CIC-IDS 2018	Binary	DL	98.87% for binary classification	The multiclass classification is not mentioned.

Quadratic map is univariate which uses a single variable. The general form of the quadratic map is shown in Equation 9.

$$f(x) = a_2x^2 + a_1x + a_0 \tag{9}$$

To match the two dimensions the gingerbreadman chaotic map to be univariate as in quadratic map, we use the general form of the hénon map shown in Equation 10 to merge the two dimensions into one dimension.

$$x_{n+1} = 1 - ax_n^2 + y_n \tag{10}$$

Hence, in this paper, we use a hybrid between three chaotic maps; ginger breadman chaotic map [43], quadratic map [46]

TABLE 2. Comparison among the studies in the reviewed literature.

Research Study	Algorithm Description	Dataset	Detection problem (Binary / Muti-class)	Classification Mode (ML / DL)	Results	Cons
[34]	K-Nearest Neighbour was the classification method employed by the proposed model. The authors used Horse herd optimization algorithm for feature selection purposes. 33 were the features that were chosen as the best. The proposed model was contrasted with a number of other algorithms in feature selection and classification, including GWO, WOA, FFA, MFO, and PSO	NSL-KDD and CSE-CIC-IDS2018	Binary	ML	99.78% for binary classification	The study did not take the multiclass classification into account
[35]	The proposed model used the classification algorithms Random Forest and Extra Trees. The authors selected features using the Arithmetic Optimization Algorithm. The best features were determined to be 43. The proposed model was efficient to extract the optimal features by combining machine learning with an altered version of the Arithmetic Optimization Algorithm	IoT traffic datasets including (NF-CSE-CIC-IDS2018-v2, NF-ToN-IoT-v2, and NF-UNSW-NB15-v2)	Both	ML	99.54% for binary and 99.52% for multiclass classifications	High model complexity
[36]	The proposed model utilized Forensics Exploratory Data Analysis as a general tool for attack investigation combined with Convolution Neural Networks. The proposed system used 72 optimal features.	UNSW-NB15, CSE-CIC-IDS2018 (CIC2018), and CIC-Darknet2020	Both	DL	98.3% for binary and 99.40% for multiclass classifications	High false positive rates
[37]	An Improved Conditional Variational Autoencoder was used in the proposed algorithm for classification. Borderline Synthetic Minority Oversampling Technique was employed by the authors in selecting features. The perfect 41 features were selected. The proposed algorithm presented IoT intrusion detection based on data augmentation to improve learning from an imbalanced samples	CSE-CIC-IDS2019	Binary	DL	98.67% for binary classification	Limited capacity to learn temporal features
[38]	The proposed algorithm used Black Widow Optimized Convolutional Long Short-Term Memory for classification. For feature selection, the authors employed Artificial Bee Colony. The perfect features were determined to be 51 features. The performance was obtained with fewer false values, less computation time, and better classification coefficients.	NSL-KDD, ISCX-IDS, UNSW-NB15, and CSE-CIC-IDS2018	Binary	DL	98.25% for binary classification	Complicated architecture

TABLE 3. List of symbols.

Symbol	Descripition
X	Zebra's population
X_i	The ith zebra in the search space
$x_{i,j}$	The value of the jth problem variable proposed by the ith zebra
N	Number of zebras
m	Number of decision variables
t	Iteration contour
X_i^{t+1}	The new status of the i-th zebra
$x_{i,j}^{t+1}$	The j^{th} dimension value for the i^{th} zebra
r	random number in the interval $[0, 1]$
PZ_j	the j-th dimension of the Pioneer Zebra (Best Solution)
$rand$	Random number in the interval $[0, 1]$
I	Random number$\in$ $1, 2$ if $I = 2$ this means more changes in the population movement
R	Constant number $= 0.01$
T_{max}	Maximum number of iterations
P_s	Probability of choosing one of the two strategies that are randomly generated in the interval $[0, 1]$
AZ_j	the jth dimension of the Attacked Zebra
c	Chaotic number which is indicated in Equation 11
R	Constant number $= 0.01$
W_{xf}, W_{xi}, W_{xc}, and W_{xo}	Input weights for forget gate, input gate, cell memory state, and output gate respectively
W_{hf}, W_{hi}, W_{hc}, and W_{ho}	Recurrent weights for hidden forget gate, hidden input gate, hidden cell memory state and hidden output gate respectively
W_{hy}	Matrix of output weights
B_f, B_i, B_c, B_o, and B_y	Bias vectors for forget gate, input gate, cell memory state and output gate respectively
NR	Number of runs
EP	Number of Epochs
HN	Number of hidden nodes
ACT	Activation function
LF	loss function
TTR	Training-testing ratio
ND	Number of nodes
CV	Cross-validation
S	Seed value
$CZOAB$	Batch size
OP	Optimizer
LR	Learning rate
DR	Dropout rate
$LSTMBS$	LSTM's Batch Size

and the hénon map [47]. Consequently, the final form given in Equation 11 represents our proposed chaotic map:

$$x_{n+2} = 1 - ax_{n+1}^2 + by_{n+1} \tag{11}$$

C. LONG SHORT-TERM MEMORY (LSTM)

Certain deep learning algorithms (DL) work better with non-sequential data than others, such as CNN and DNN. They are capable of processing multimedia content. But for time series, it is better to use Recurrent Neural Networks (RNNs) to analyze patterns. They can be applied to activities such as speech recognition, image-to-text conversion, and video event detection [5]. Long Short-Term Memory in RNNs is built on repeating operations related to contextual information in hidden layers. Furthermore, gradient vanishing and explosion issues prevent RNNs from serving as LSTM [48].

Algorithm 1 Zebra Optimization Algorithm

```
Set the population size (N), number of decision variables (m), and maximum number of iterations (T_max)
Initialize the positions of the zebra population randomly
Evaluate the objective function of zebras
t=0
while t ≤ T_max do
    for i = 1 to N do
        for j = 1 to m do
            Phase 1: Foraging Behavior
            Calculate the new position of the j^th Pioneer Zebra PZ_j
            Calculate the new position of the i^th zebra x_i^{t+1} using Equation 4
            Update the position of the i^th solution X_i^t using Equation 3
            Phase 2: Defense strategies against predators
            Compute P_s = rand
            if P_s ≤ 0.5 then
                Strategy 1: Zebra against lion (Exploitation Phase)
                Calculate the new position of the i^th zebra x_i^{t+1} using mode S1 in Equation 7
            else
                Strategy 2: Zebra against other predators (Exploration Phase)
                Calculate the new position of the j^th Attacked Zebra AZ_(j
                Calculate the new position of the i^th zebra x_i^{t+1} using mode S2 in Equation 7
            end if
            Update the position of the i^th solution X_i^t using Equation 6
        end for
    end for
    t=t+1
    Save the overall best solution
end while
Output: the optimal solution obtained by ZOA
```

Therefore, Hochreiter and Schmidhuber presented a Long Short-Term Memory network (LSTM) for time-series prediction in 1997 [49]. When memory cells are added to the hidden layer, the LSTM can govern how time-series data is stored in memory. A set of programmable gates (input, output, and forget gate) are used to transmit data between the cells in the hidden layers [50]. The vanishing gradient and explosion problems are avoided because LSTM can preserve the cell state through its gate mechanism, which can resolve both short-term and long-term memory dependency issues [51].

The primary structure of the proposed LSTM algorithm for CZOLSTM is shortened to Algorithm 2. The basic LSTM cell

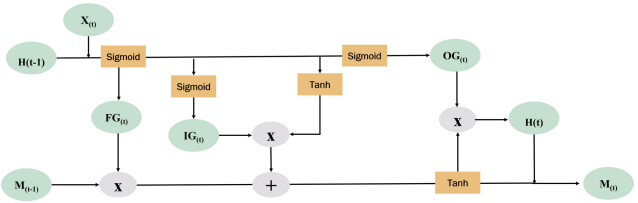


FIGURE 1. The block diagram of LSTM cell.

in a memory cell has three gates - input gate, output gate, and forget gate - which is depicted in Figure 1, which was previously explained in [51]. The input gate represented by the symbol *IG* is in charge of monitoring the most recent data within a memory cell. The value of the input gate at the time instance t is expressed in Equation 12.

$$IG(t) = sigmoid(W_{xi}X(t) + W_{hi}H(t-1) + B_i) \tag{12}$$

The output gate manages the distribution of the most recent data to other networks. It is denoted by the symbol *OG* and the output gate's value at time instance t is expressed in Equation 13.

$$OG(t) = sigmoid(W_{xo}X(t) + W_{ho}H(t-1) + B_o) \tag{13}$$

The forget gate determines whether or not the data should be erased based on the condition of the previous cell. It is denoted by the symbol *FG* and the forget gate's value at time instance t is expressed in Equation 14.

$$FG(t) = sigmoid(W_{xf}X(t) + W_{hf}H(t-1) + B_f) \tag{14}$$

The hidden state depends on the cell's memory state. It is denoted by the symbol $H(t)$ and the hidden state's value at the time instance t is expressed in Equation 15.

$$H(t) = OG(t)\tanh(M(t)) \tag{15}$$

The cell memory state is denoted as $M(t)$ and its value at the time instance t is expressed in Equation 16.

$$M(t) = FG(t) \times M(t-1) + IG(t) \times (sigmoid(W_{xc}X(t) + W_{hc}H(t-1) + B_c)) \tag{16}$$

The sigmoid function is expressed in Equation 17.

$$sigmoid(x) = \frac{1}{1+\exp^{-x}} \tag{17}$$

where X denotes the input vector, the weight matrices W_{xf}, W_{xi}, W_{xc}, and W_{xo} represent input weights, while W_{hf}, W_{hi}, W_{hc}, and W_{ho} represent recurrent weights. W_{hy} matrix of output weights; with the corresponding bias vectors B_f, B_i, B_c, B_o, and B_y.

Algorithm 2 LSTM Algorithm

```
Initialize parameters: LSTMBS, Epoch, Input shape (I_{0,d}) and Load data instances from dataset
for Run = 1 to the number of runs (NR) do
    Define sequential model
    for Epoch = 1 to the number of epochs (EP) do
        Detect and classify attacks using a fully connected Dense layer with HN and ACT function
        Compile the model using OP and LF
        Fit the model on the TTR and LSTMBS
        Evaluate and predict model for the testing set
    end for
end for
Obtain the overall best solution
```

Where *NR* is the number of runs, *EP* is the number of Epochs, *HN* is the number of hidden nodes, *ACT* represents the activation function, *OP* is the optimizer, *LF* is the loss function, *TTR* is the training-testing ratio, and *LSTMBS* is the batch size for LSTM. The assigned values for these parameters are given in Table 6

D. THE PROPOSED CHAOTIC ZEBRA OPTIMIZATION LONG SHORT-TERM MEMORY ALGORITHM (CZOLSTM)

CZOLSTM is considered a new hybrid chaotic variant of the swarm intelligence Zebra Optimization Algorithm. Swarm intelligence optimization algorithms rely on pseudo-random numbers. Chaotic maps were frequently used in place of the pseudo-random numbers that were distributed in the Gauss distribution in computer engineering. The chaotic optimization method was named after these optimization advancements [27]. To improve chaotic qualities for a wide range of system parameters, a new chaotic system must be developed because chaotic maps are computationally expensive and difficult to apply [28]. As mentioned in Section III-B, the hybrid chaotic map combines the special features of Hénon's map, Gingerbread Man, and Quadratic. With this hybrid map, a rich, diverse, and unexpected chaotic structure is generated. A comparison was made between this chaotic ZOA and the standard ZOA.

While a number of optimization problems have been effectively resolved by the ZOA. There are several effects it can have, including early convergence, speed in some difficult scenarios, and calculation time [52]. Therefore, recommendations for modifications were made to address them. CZOLSTM is an improved variant of the ZOA algorithm using chaotic maps. It concentrates on boosting the speed and the balance between exploitation and exploration. Despite being new, the ZOA algorithm has an advantage over all others in that it requires fewer parameters.

Therefore, employing chaotic maps rather than the randomization parameter I could improve the accuracy of the ZOA [52]. Since the randomization setting ignores more significant changes in population movement, the chaotic structure's generation is unexpected [53]. The primary

structure of the standard ZOA remains unchanged, however the equations went through some changes.

For global search, Equation 4 can be changed as follows:

$$x_{i,j}^{t+1} = x_{i,j}^{t} + C.(PZ_j - I \times x_{i,j}^{t}) \tag{18}$$

$$I = round(1 + C) \tag{19}$$

where $x_{i,j}^{t+1}$ is the j^{th} dimension value for the i^{th} zebra, PZ_j is the jth dimension of the Pioneer Zebra (Best Solution), I is a number obtained by Equation 19 and C is a chaotic number which is indicated in Equation 11.

While for local search, Equation 7 can be updated as follows:

$$x_{i,j}^{t+1} = \begin{cases} S1 : x_{i,j}^{t} + R.(2C-1).(1 - \dfrac{t}{T_{max}}).x_{i,j}^{t}, \\ \quad \text{if } P_s <= 0.5. \\ S2 : x_{i,j}^{t} + C.(AZ_j - I \times x_{i,j}^{t}), \\ \quad \text{otherwise.} \end{cases} \tag{20}$$

where X_i^{t+1} is the new status of the ith zebra based on the second phase $P2$, R is constant number $= 0.01$, r is random number in the interval [0, 1], t is the iteration contour, T_{max} is the maximum number of iterations, P_s is the probability of choosing one of the two strategies that are randomly generated in the interval [0, 1], AZ_j is the jth dimension of the Attacked Zebra, I is a number obtained by Equation 19, and C is a chaotic number which is indicated in Equation 11. Algorithm 3 presents the steps of the CZOLSTM algorithm.

Algorithm 3 CZOLSTM Algorithm

```
Load data instances from the dataset
Apply data preprocessing as explained in Section III-D1
Set the population size (N), number of decision variables (m),
  and maximum number of iterations (T_max)
Initialize the positions of the zebra population randomly
Evaluate the objective function of zebras
t=0
while t ≤ T_max do
    for i = 1 to N do
        for j = 1 to m do
            Phase 1: Foraging Behavior
            Calculate the new position of the j^th Pioneer
            Zebra PZ_j
            Calculate the new position of the i^th zebra
            x_i^{t+1} using Equation 18
            Update the position of the i^th solution X_i^t
            using Equation 3
            Phase 2: Defense strategies against
            predators
            Compute P_s = rand
            if P_s ≤ 0.5 then
                Strategy 1: Zebra against lion
                (Exploitation Phase)
                Calculate the new position of the
                i^th zebra x_i^{t+1} using mode S1
                in Equation 20
            else
                Strategy 2: Zebra against other
                predators (Exploration Phase)
                Calculate the new position of
                the j^th Attacked Zebra AZ_j
                Calculate the new position of
                the i^th zebra x_i^{t+1} using
                mode S2 in Equation 20
            end if
            Update the position of the i^th
            solution X_i^t using Equation 6
        end for
    end for
    t=t+1
    Save the best candidate set of selected
    features
end while
Classification Phase
Apply LSTM as shown in Algorithm 2
Output: The overall best solution
```

1) DATA PREPROCESSING

The files in the CSE-CIC-IDS-2018 dataset contain several data instances from several attacks. This makes processing the data samples stored there and merging them to include each attack label is extremely time-consuming and computationally challenging. Moreover, there is a high-class imbalance in the CIC-IDS-2017 and CSE-CIC-IDS-2018 datasets, which could indicate the system's poor accuracy and high false positive rate. For this reason, the data preprocessing phase is quite important. Data preprocessing is responsible for eliminating null, missing values, duplicate, and inaccurate records. Also, it can solve the issue of class imbalance. The process of preparing the data for use in our proposed algorithm could be explained in the following lines:

- Elimination of NAN and infinite values: Any null, missing, or duplicate values are found and eliminated during this phase.
- Label encoding: In this phase, every data instance related to a particular attack is separated. The data associated with each attack is then converted into a numerical matrix representation using label encoding. The target labels for each attack are additionally encoded with values ranging from zero to the number of classes minus one.
- Data shuffling/re-sampling:
  There is a class imbalance problem in the CIC-CSE IDS 2018 dataset. Unbalanced datasets have a substantial impact on the distribution of classes. Classification models tend to overclassify the larger class due to its higher prior probability. Thus, instances of the smaller class are more likely than the bigger class to be incorrectly classified. This stage attempts to deal with the issue of class imbalance. The only ways to enhance the learning process are by eliminating bias from the data and by providing equal distributions. Because overfitting happens when one class has more samples than the others, resampling is done to balance the data distribution across all attack categories by providing equal volumes of data from various attack categories.

- Data normalization: This is the process of rescaling data from its original range to a new range ranging from zero to one where all values fall. Consequently, all features' values have been scaled using the min-max normalization approach to fall between zero and one. All values are divided by the maximum value that is encountered, and all values are divided by the range between the maximum and minimum values after the minimum value is eliminated. Assuming that x represents the feature value that requires normalization, y represents the normalized value of x, and that the lowest and highest values within the real range are min and max. Equation 21 denotes how the x value is normalized.

$$y = \frac{x - min}{max - min} \tag{21}$$

2) THE STRUCTURE OF THE PROPOSED CZOLSTM

The main structure of the proposed CZOLSTM algorithm for enhancing cyber security threat identification is displayed in Algorithm 3. One significant benefit is that it makes use of both LSTM as a deep learning algorithm for cyber threat classification and CZOA as a chaotic novel swarm intelligence algorithm used for feature selection. Preprocessing the data, selecting the appropriate features, classifying the data, training and validating the algorithm, and lastly conducting testing and evaluation are the five main phases of the proposed algorithm.

The main steps of the proposed algorithm can be summarized as follows:

- **Step 1**, The CZOLSTM settings are initially initialized with their default values. The hyperparameters of the CZOLSTM algorithm were then optimized. The optimized values are listed in Table 6. The justification for selecting these values is discussed in detail in Section IV-D.
- **Step 2**, The accuracy and high false positive rate of the system are impacted by the substantial class imbalance presented in the CSE-CIC-IDS-2018 dataset. Processing massive data instances is computationally difficult and time-consuming. Thus, data preprocessing is essential. It addresses the problem of class imbalance by removing bias or redundant data and offering equal distributions. The details of this step are demonstrated in Section III-D1.
- **Step 3**: Every zebra's position symbolizes a potential solution to the feature selection problem. In this step, the population size N is set to be the number of zebras in the search space, m is the number of decision variables, and the maximum number of iterations T_{max}.
- **Step 4**: Since each zebra is considered a solution so in our case each zebra represents some specified feature. In this step, the positions of the zebra population are randomly selected.
- **Step 5**: Consequently, the suggested values of each zebra for the problem variables can be used to evaluate the objective function. In this step, the objective function of the zebras (features) is evaluated.
- **Step 7** The following steps are repeated until the termination criterion is satisfied.
- **Step 11**: At each iteration t, the position of the j^{th} Pioneer Zebra (local best solution) PZ_j.
- **Step 12**: The position of each zebra is modified as indicated in Equation 18.
- **Step 13**: The objective function of each solution in the population is evaluated (F_i) and based on its value, the position of the i^{th} solution X_i^t is updated using Equation 3
- **Step 15**: Assign a random number for P_s which is the probability of choosing one of the two strategies that are randomly generated in the interval [0, 1].
- **Step 18**: based on the value of P_s if it was less than or equal to 0.5, Then, the new position of the i^{th} zebra x_i^{t+1} is updated based on Strategy 1 using Equation 20
- Otherwise, In **Step 21**: the position of the j^{th} Attacked Zebra AZ_j is updated. In **Step 22**: the new position of the i^{th} zebra x_i^{t+1} is updated based on Strategy 2 using Equation 20
- **Step 24**: the position of the i^{th} solution X_i^t is updated using Equation 6.
- **Step 27**: The operation is repeated until the termination criteria are satisfied.
- **Step 28**: The best candidate set of selected features is recorded.
- **Step 31**: In this step, a deep LSTM network comprising two input and output layers and three hidden layers is applied. It transfers an input layer to their representations using the optimal selected features as an input layer. The sequence feeds the one-dimensional convolutional layer. Max pooling layers receive the convolution layer's output after Tanh is applied as an activation function. After that, the output is moved to the LSTM layer, where a dropout rate of 0.1 is used. Finally, the data collected from the LSTM layer is acquired and aggregated by fully linked layers. The final result depends on the tested classification modes (binary or multiclass). Data splicing produces training and testing sets. Training data serves to train the DL algorithm, whereas testing data evaluates the model's performance on untested data subsets. The data is split into a ratio 80-20. This means that 80 percent of the data was utilized for training and the remaining 20 percent was used for testing. The evaluation measures described in Section IV-C are then used to evaluate performance.
- **Step 32:** Produce the best found solution so far.

3) TIME COMPLEXITY OF THE PROPOSED CCOLSTM ALGORITHM

The time complexity of the CCOLSTM algorithm can be computed as follow:

a: INITIAL POPULATION

The time complexity of the initial population is $O(N \times m)$, Where N is the population size and m is the problem dimension.

b: SOLUTION UPDATE

The time complexity for all solutions in the population is $O(N \times m)$.

c: FITNESS FUNCTION EVALUATION

The time complexity to calculate the fitness function of all solutions in the population is $O(N \times m)$.

d: LSTM ALGORITHM

$NR \times EP \times ND$, where NR is the number of runs, EP is the number of epochs, ND is the number of nodes.

The total time complexity is $O(N \times D \times T_{max}) + O(NR \times EP \times ND)$, T_{max} is the maximum number of iterations.

IV. EXPERIMENTS AND DISCUSSIONS

In order to validate the experimental findings, additional tests with different parameters were carried out. Additionally, a comparison is made between the results of the standard LSTM with all features and the proposed CZOLSTM algorithm with the selected features. Additionally, a variety of binary and multiclass classification modes were examined. This section presents the experimental setup and findings. The description of the hardware and software configuration is given in Section IV-A. The CSE-CIC-IDS2018 Dataset, which is detailed in Section IV-B, was utilized to test the proposed CZOLSTM. Performance is assessed using the evaluation metrics outlined in Section IV-C. The dataset was preprocessed and the parameters of the proposed algorithm were adjusted as explained in Section IV-D. In the initial setting of the standard LSTM, all 79 features of the dataset are input into the model. However, in subsequent tests, we employed CZOA as a feature selection algorithm to reduce the total number of features and enhance the performance that was obtained. Section IV-E describes how CZOLSTM can be used efficiently for feature selection. Both binary and multiclass classifications were used to assess the proposed CZOLSTM algorithm. Section IV-F discusses the efficiency of the proposed CZOLSTM for binary classification, whereas Section IV-G discusses the efficiency of the proposed CZOLSTM for multiclass classification. Both sections cover a detailed description of the performance obtained from each classification mode, the confusion matrix (CM) generated from the classification, and comparisons with other similar studies.

A. SYSTEM CONFIGURATION

For speeding up the process of DL computation, a GPU working with NVIDIA CUDA (Compute Unified Device Architecture) is used to reduce computation times and possibly meet real-time data processing needs. Using a Kaggle Platform instance, we run all of our studies on NVIDIA P100 and NVIDIA T4(x2) GPUs. We used Keras with a Tensorflow backend as our deep learning framework. The NVIDIA P100 has Pascal architecture, 16GB of RAM, 9.5 TeraFLOP/s of Single Precision FLOPs, and 732 GB/s of memory bandwidth. NVIDIA T4, on the other hand, has 16GB of memory, 8.1 TeraFLOP/s of Single Precision FLOP/s, and 320 GB/s of memory bandwidth. For CPU tasks, the 16GB RAM allocation is raised to 30GB per session. The hardware and software specifications used in the experiments are listed in Table 4.

B. CSE-CIC-IDS2018 DATASET

The CSE-CIC-IDS2018 intrusion detection dataset was created in 2018 by the Communications Security Establishment and the Canadian Institute for Cybersecurity, both of which have their headquarters in Fredericton, Canada [54], [55]. It is the most recent intrusion dataset, acquired to conduct real attacks [56]. It is available to the public. CSE-CIC-IDS2018 has a larger capacity than CICIDS2017, with over 400GB. The dataset includes both the attack dataset's required standards and a broad range of attacks. It primarily comprises seven distinct attack scenarios: distributed denial-of-service, botnet, denial-of-service, brute force, heartbleed, web attacks, and network infiltration [55]. A dataset was created using HTTP, HTTPS, FTP, SSH, and email protocols to simulate the online activities of 25 fake users. The sub-datasets that make up CIC-2018 were collected over a period of 10 different days, following the injection of 16 distinct forms of attack.

There are several attacks, including SSH-BruteForce, FTP-BruteForce, Brute Force -XSS attack, Brute Force -Web, SQL Injection, DoS attacks using Hulk, SlowHTTPPTest, Slowloris, DoS attacks using GoldenEye, DDOS attacks using HOIC, DDOS attacks using LOIC-UDP, DDOS attacks using LOIC-HTTP, and more. Common attacks are listed in Table 5. To more accurately represent the attacks, a network infrastructure with machine diversity similar to real-world networks was developed (five departments making up the victim organization, with fifty attacker machines, four hundred victim machines, and thirty servers) [56], [3]. The dataset includes 80 features that CICFlowMeter-V3 retrieved from the traffic, together with forward and backward collected system logs and network traffic for each machine [57]. Ten files totaling 15, 450, 706 rows, each with 80 characteristics, make up the CSE-CIC-IDS2018 dataset [58].

C. EVALUATION METRICS

Common performance metrics were used to evaluate the performance including accuracy, precision, recall, AUC value, and F1 which are used to evaluate classifiers based on our proposed algorithm [59]. Furthermore, trials were conducted extensively to distinguish between malicious and legitimate data instances. Furthermore, the confusion matrix

TABLE 4. List of hardware resources and software environments.

Software specification	Software tool	Jupyter notebook
	Programming language	Python 3.7.12
	API tools	Keras Sklearn library Matplotlib 3.5
	DL framework	TensorFlow
	Computing platform	Kaggle Platform
Hardware Specification	Processor	Intel® Core(TM) i5-5200U
	CPU	2.20 GHZ
	RAM	6 GB
	OS	64-bit Windows 8.1
	GPU	NVIDIA P100 and NVIDIA T4(x2) GPUs

TABLE 5. Labels for CSE-CICIDS2018 Attacks.

Label	Attack name	Number of Instances
1	FTP-BruteForce	35127
2	SSH-Bruteforce	33817
3	DDOS-HOIC	124280
4	Bot	52051
5	DoS-GoldenEye	7598
6	DoS-Slowloris	2028
7	DDOS-LOIC-UDP	3019
8	Brute Force-Web	9108
9	Brute Force-XSS	4632
10	SQL Injection	53

TABLE 6. Adjusted hyperparameters of the proposed CZOLSTM.

Method	Parameter	Definition	Value
CZOA	CV	Cross-validation	10 folds
	T_max	Max iterations	50
	R	Constant number	0.01
	N	Population size	25
	R	R value for CZOA	0.01
	S	Seed value	1234
	CZOAB	Batch size	64
LSTM	TTR	Training-testing ratio	80%-20%
	OP	Optimizer	Adam
	LR	Learning rate	0.001
	DR	Dropout rate	0.1
	LSTMBS	Batch size	32
	NR	Number of Runs	20
	EP	Epochs	50
	HN	Hidden Nodes	64
	ACT	Activation function	Tanh
	LF	Loss function	Mean Squared Error

(CM) was used to compute the performance metrics in our work [60], [61]. The following acronyms can be used to refer to CM:

- True Positives (TP): The model accurately categorizes benign events.
- True Negatives (TN): The model accurately identifies malicious attempts.
- False Positives (FP): In all observations, anomalies are incorrectly expected to be normal occurrences. This number should ideally have a low value.
- False Negatives (FN): Malicious attacks that are incorrectly classified by the model as benignÂ occurrences. This number should ideally have a low value.

The proposed algorithm was assessed using the following metrics:

- Accuracy is the percentage of all expected occurrences, whether normal or abnormal, that were correctly predicted from all observations. This metric, which is commonly used to evaluate model performance, is especially useful when the classes are not balanced. Its value is calculated using the Equation 22.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{22}$$

- Precision is the number of accurately predicted positive observations relative to all expected positive observations. Low false positive rates are negatively correlated with accuracy. Greater precision levels correspond to better results. Its value is calculated using the Equation 23.

$$Precision = \frac{TP}{TP + FP} \tag{23}$$

- Recall is the ratio of the correctly predicted positive observations relative to all observations. It should be as high as possible. Its value is calculated using the Equation 24.

$$Recall = \frac{TP}{TP + FN} \tag{24}$$

- F1 score is produced by harmonically averaging precision and recall. Its value is calculated using Equation 25.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{25}$$

- Area Under the Curve (AUC) indicates how effectively a machine learning model matches our expectations for identifying or classifying various scenario types [62]. It indicates the probability that a positive sample will outnumber a negative sample when the rating is taken into consideration. Its value is calculated using the

TABLE 7. The adjustment of the dropout rate for the CZOLSTM algorithm.

Adjusted Parameters			Performance metrics					
Dropout	Hidden nodes	Activation Function	Accuracy	Loss	F1	Precision	Recall	AUC
0.0	64	relu	94	0.01	97.82	97.88	97.83	98.81
0.1	**64**	**relu**	**95**	**0.01**	**97.38**	**97.74**	**97.07**	**98.42**
0.2	64	relu	93	0.01	96.78	97.32	96.47	98.1
0.3	64	relu	89	0.02	87.93	97.66	83.68	91.75
0.4	64	relu	88	0.02	89.98	95.06	87.6	93.49
0.5	64	relu	90	0.02	96.53	97.85	95.44	97.61
0.6	64	relu	86	0.03	86.55	98.54	81.45	90.66
0.7	64	relu	85	0.04	92.07	96.88	89.62	94.65
0.8	64	relu	78	0.04	85.9	99.35	79.12	89.53
0.9	64	relu	60	0.06	27.33	36.36	27.3	63.65

TABLE 8. The adjustment of the hidden nodes for the CZOLSTM algorithm.

Adjusted Parameters			Performance metrics					
Dropout	Hidden nodes	Activation Function	Accuracy	Loss	F1	Precision	Recall	AUC
0.1	8	relu	61.69	0.05	41.41	62.94	37.53	68.74
0.1	16	relu	77.59	0.03	69.32	86.11	61.05	80.41
0.1	32	relu	90.93	0.02	94.35	95.3	93.98	96.74
0.1	**64**	**relu**	**97.74**	**0.01**	**96.04**	**97.02**	**95.36**	**97.53**
0.1	128	relu	96.55	0.01	97.61	97.74	97.54	98.66

TABLE 9. The adjustment of the activation function for the CZOLSTM algorithm ranked by the best performance.

Adjusted Parameters			Performance metrics					
Dropout	Hidden nodes	Activation Function	Accuracy	Loss	F1	Precision	Recall	AUC
0.1	**64**	**tanh**	**98.23**	**0**	**0.994711**	**0.994847**	**0.994643**	**0.997061**
0.1	64	sigmoid	97.85	0	0.994694	0.99507	0.994383	0.996942
0.1	64	softmax	97.15	0.01	0.984358	0.986511	0.982971	0.990785
0.1	64	linear	95.62	0.01	0.979264	0.990119	0.969471	0.984243
0.1	64	relu	93.92	0.01	0.958432	0.964293	0.955074	0.975615

TABLE 10. The efficiency of performance metrics through a different number of epochs of the proposed algorithm based on binary classification.

# Epochs	# Runs	Accuracy	loss	F1	Precision	Recall	AUC
20	5	97.78	0.01	97.73	97.88	97.65	98.72
30	5	98.65	0	98	98.6	97.6	98.7
	10	98.95	0	99.2	99.4	99.2	99.5
50	**20**	**98.98**	**0**	**99.4**	**99.4**	**99.4**	**99.7**

Equation 26.

$$AUC = \frac{n}{P.N} \tag{26}$$

D. PARAMETER SETTING

Table 6 displays the optimized hyperparameters for the CZOLSTM algorithm. Using the Adam optimizer, the trainable parameter was changed at a learning rate of 0.001. Furthermore, it's clear that performance changes gradually after using optimized values for each hyperparameter from using the default hyperparameters. Thus, using the default hyperparameters produces an accuracy of 90%. Furthermore, the accuracy increased to 95.23% after using a dropout rate of 0.1. After adjusting the hidden nodes to 64, the accuracy raised to be 96.74%. Finally, the accuracy raised to 98.23% after using tanh as an activation function. The results of several in-depth tests conducted with various hyperparameters for the CZOLSTM algorithm are shown in Tables 7, 8, and 9 in where the overall best values are reported in bold text.

Setting the different parameters was harder than it seemed at first. At this point, we decided not to use any algorithms for hyperparameter optimization. However, when the outcomes weren't satisfactory, we proceeded step-by-step to manual adjustment. Using the default hyperparameter settings and without making any parameter adjustments, the dataset was split 80-20 into 80% and 20% training and testing sections, respectively. For a fair evaluation, five cross-validation splits are performed, and the mean is computed. The obtained accuracy was 97.78 percent, the loss was 0.01; the precision was 97.88; the recall was 97.65; and the AUC was 98.72. However, in contrast to other comparable research, that was very low.

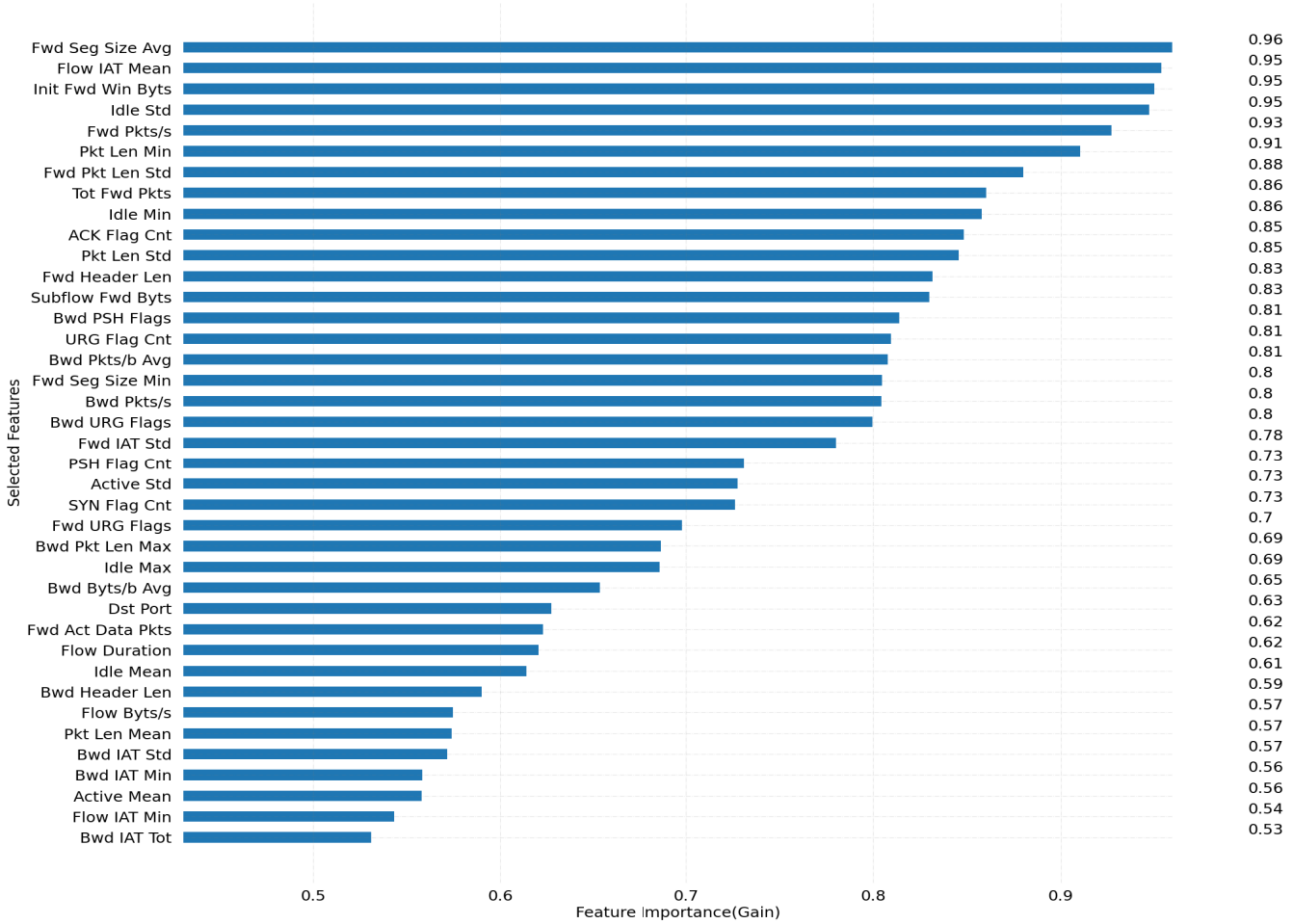


FIGURE 2. Feature importance using CZOLSTM.

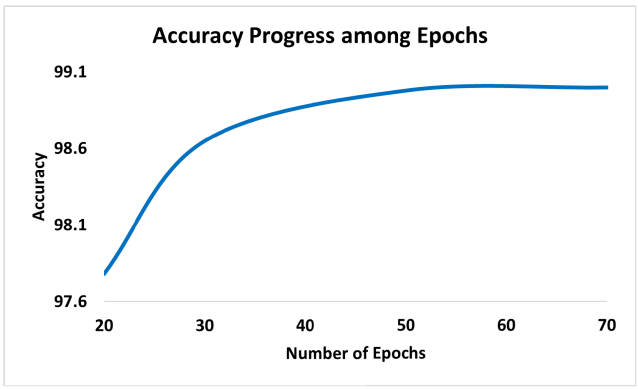


FIGURE 3. The accuracy progress among different number of epochs.

In this step, the hyperparameters of the proposed algorithm were manually adjusted in an attempt to improve performance. For five cross-validation splits, the proposed algorithm was evaluated using the same performance metrics. To determine whether changing the hyperparameters affected the CZOLSTM algorithm's performance, the recommended algorithm's default settings were progressively changed for thirty epochs in each run.

The accuracy of the system is influenced by various parameters, such as the activation function, number of hidden nodes, and dropout rate. We proceed with modifying each of these elements separately. The most appropriate parameters for evaluating the CZOLSTM were identified after several runs, and the outcomes were contrasted with those of the earlier experiments.

1) THE ADJUSTMENT OF THE DROPOUT RATE

As seen in Table 7, the CZOLSTM algorithm's dropout rate is updated progressively from 0.1 to 0.9. In each trial, the dropout rate increases by 0.1 and uses fixed values for hidden nodes of 64 and the relu as an activation function. Bold text reports the overall best values. It has been observed that accuracy increases from 60% to 95% when the dropout rate is decreased from 0.9 to 0.1. The results clearly show that the ideal value for dropout is 0.1, yielding the maximum accuracy of 95%.

2) THE ADJUSTMENT OF THE HIDDEN NODES

The number of hidden nodes for the CZOLSTM algorithm is adjusted consecutively from 8 to 128 while maintaining stable

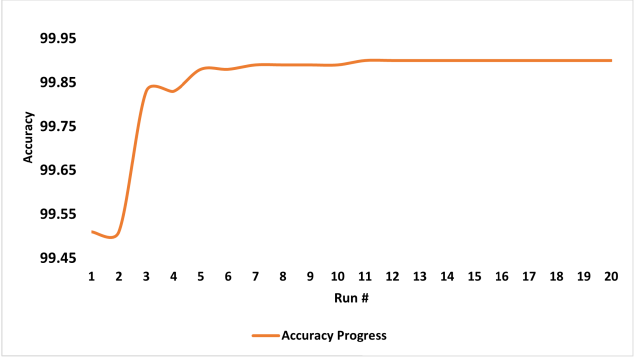


FIGURE 4. Accuracy progress for CZOLSTM.

values for a dropout rate of 0.1, as it was the ideal value. To perform this update, each trial's hidden node count is doubled. The relu served as an activation function throughout these upgrades. The performance changes while updating the hidden nodes are shown in Table 7. Increasing the number of hidden nodes from 8 to 64 has been found to increase accuracy from 61.69% to 97.74%. Thus, 64 is the best value for hidden nodes, and when the other parameters are fixed, this gives the highest accuracy of 97.74%.

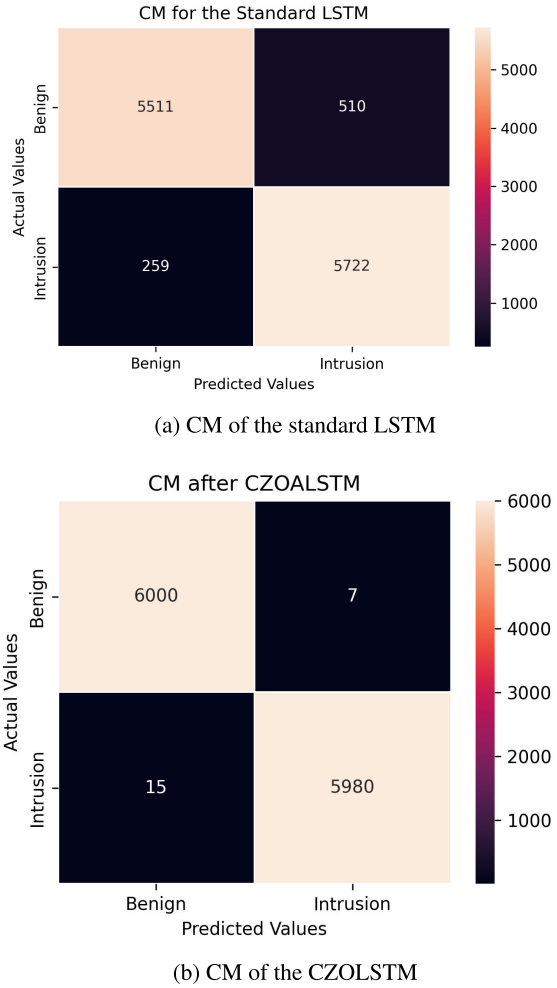
3) THE ADJUSTMENT OF THE ACTIVATION FUNCTION

For each test, the activation function for the CZOLSTM algorithm is updated using a combination of tanh, sigmoid, softmax, linear, and relu. The dropout rate of 0.1 and the number of hidden nodes of 64 were fixed because these were the best values, as indicated in Tables 7 and 8, respectively. Based on the best results obtained, Table 9 displays the performance progress in order during updates. It is observed that increasing the accuracy from 93.92% to 98.23% occurs when the activation function is changed from relu to tanh. It's clearly shown that, while other parameters are fixed, tanh is the optimal choice for the activation function, yielding the highest accuracy of 98.23%.

4) THE ADJUSTMENT OF THE EPOCHS

After selecting the best settings, the next test was advanced by increasing the number of epochs in each iteration. The epoch count increased from 30 to 50, and the number of runs increased from 5 to 20. The top-performing results overall are shown in Table 10. This table shows the improvement in the binary classification performance for CZOLSTM based on Accuracy, Loss, F1, Precision, Recall, and AUC throughout several runs and epochs. Figure 3 depicts the accuracy progress among different numbers of epochs. Performance is enhanced by increasing the number of runs and epochs. When the number of epochs and runs is increased, the accuracy averages out at 98.98 instead of 98.23 as obtained after adjusting parameters.

Prior to FS, the CZOLSTM was assessed and yielded the following results: 97.78% accuracy, 97.73 F1, 97.88 precision, 97.65 recall, and 98.72 AUC. However, the accuracy increased to 98.65%, the precision to 98.6, the recall to 97.6,



(a) CM of the standard LSTM

(b) CM of the CZOLSTM

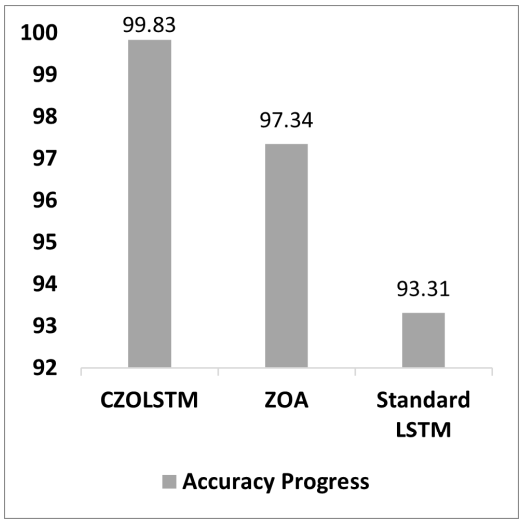
FIGURE 5. CM of the standard LSTM and the CZOLSTM based on binary classification.

the AUC to 98.7, and there was no loss. This is achieved by using five different runs, thirty epochs, and the adjusted hyperparameters.

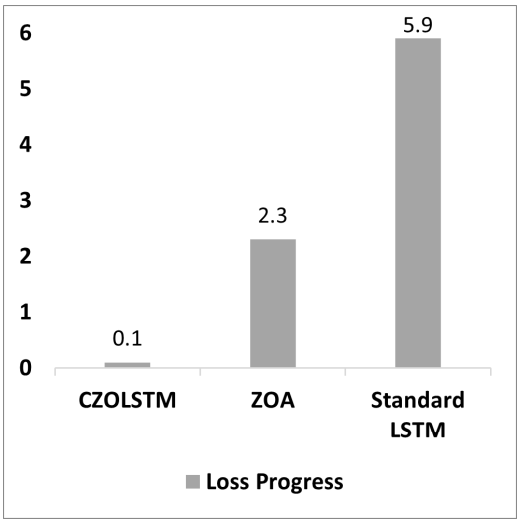
After adjusting the hyperparameters to 30 epochs and 10 distinct runs, the accuracy reached 98.95%, the F1 was 99.4, the precision was 99.4, the recall was 99.4, the AUC was 99.7, and there was no loss. The results demonstrated 98.98% accuracy, 99.2 F1, 99.4 precision, 99.2 recall, 99.5 AUC, and zero loss using 50 epochs and 20 separate runs. As seen in Figure 3, a larger number of epochs can improve the outcomes. However, there will be a rise in computation and time complexity. As a result, deciding on fifty epochs as the ideal number will strike a compromise between evaluated performance and time complexity.

E. THE EFFICIENCY OF CZOLSTM FOR FEATURE SELECTION

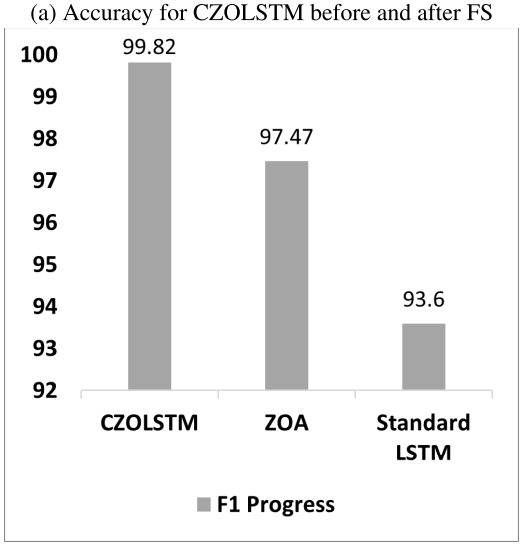
The best performance would come from not utilizing all of the features in the dataset because some of them were redundant and unnecessary. This provides motivation to investigate if using optimal features has an



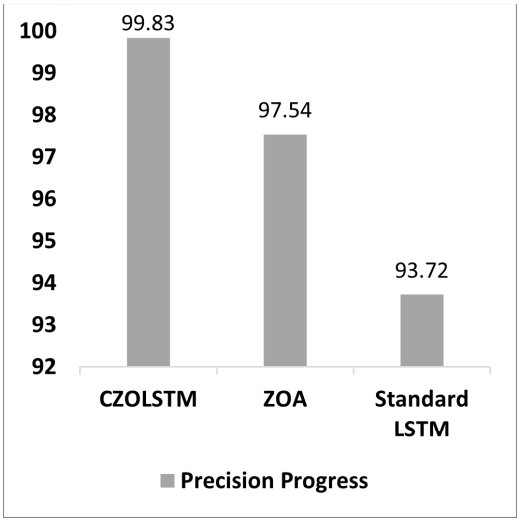
(a) Accuracy for CZOLSTM before and after FS



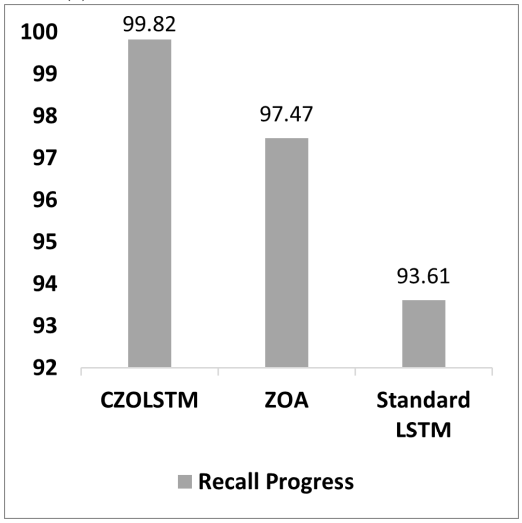
(b) Loss for CZOLSTM before and after FS



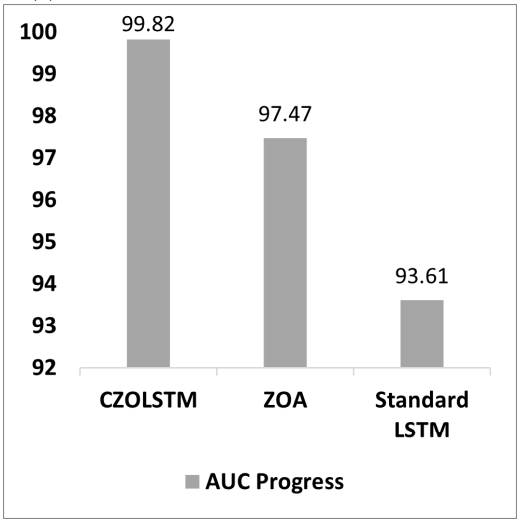
(c) F1 for CZOLSTM before and after FS



(d) Precision for CZOLSTM before and after FS



(e) Recall for CZOLSTM before and after FS



(f) AUC for CZOLSTM before and after FS

FIGURE 6. **Comparison between CZOLSTM, ZOA and the standard LSTM in performance based on binary classification.**

TABLE 11. Overview of the selected features based on CZOLSTM feature selection algorithm.

Fwd Seg Size Avg	Flow IAT Mean	Init Fwd Win Byts	Idle Std	Fwd Pkts/s	Pkt Len Min
Fwd Pkt Len Std	Tot Fwd Pkts	Idle Min	ACK Flag Cnt	Pkt Len Std	Fwd Header Len
Subflow Fwd Byts	Bwd PSH Flags	URG Flag Cnt	Bwd Pkts/b Avg	Fwd Seg Size Min	Bwd Pkts/s
Bwd URG Flags	Fwd IAT Std	PSH Flag Cnt	Active Std	SYN Flag Cnt	Fwd URG Flags
Bwd Pkt Len Max	Idle Max	Bwd Byts/b Avg	Dst Port	Fwd Act Data Pkts	Flow Duration
Idle Mean	Bwd Header Len	Flow Byts/s	Pkt Len Mean	Bwd IAT Std	Bwd IAT Min
Active Mean	Flow IAT Min	Bwd IAT Tot			

TABLE 12. Efficiency of the proposed CZOLSTM for binary Classification.

Algorithm	Accuracy	Loss	F1	Precision	Recall	AUC
The proposed CZOLSTM	**99.83**	**0.1**	**99.82**	**99.83**	**99.82**	**99.82**
ZOA	97.34	2.3	97.47	97.54	97.47	97.47
Standard LSTM	93.31	5.9	93.6	93.72	93.61	93.61

TABLE 13. Comparsion between the CZOLSTM and Other Related Algorithms Based on Binary Classification.

Study	Accuracy	F1	Precision	Recall
BWO-CONV-LSTM [38]	98.25	98.18	97.48	98.67
NAID [63]	98.85	98.83	98.85	98.85
IMBO [33]	98.87	97.84	97.48	98.18
MLP-PSO [30]	98.97	99.38	**99.98**	98.8
FEDA [36]	99.4	98.5	94	89
AOA [35]	99.54	99.54	99.54	NA
The Proposed CZOLSTM	**99.83**	**99.82**	99.83	**99.82**

impact on performance. In order to improve the result and decrease the amount of features, we thus used the feature selection algorithm. The most important features that the CZOLSTM algorithm effectively selected are represented by Figure2 where the x-axis represents the feature's importance (Gain) and the y-axis is the associated feature.

The optimal features were successfully selected using the CZOA FS algorithm, which eliminated the weaker ones. A thorough explanation of the main algorithm used by CZOLSTM may be found in Section III-D. Just 39 out of the 79 features in the original dataset were selected using the CZOA FS algorithm. Using the 39 optimal features given in Table 11, the proposed CZOLSTM algorithm was further improved.

The feature selection process went through multiple iterations. At each iteration, less important features were removed. In total, there are 79 features. The feature importance of the 39 significant features that were selected by the CZOA FS algorithm is shown in Figure 2.

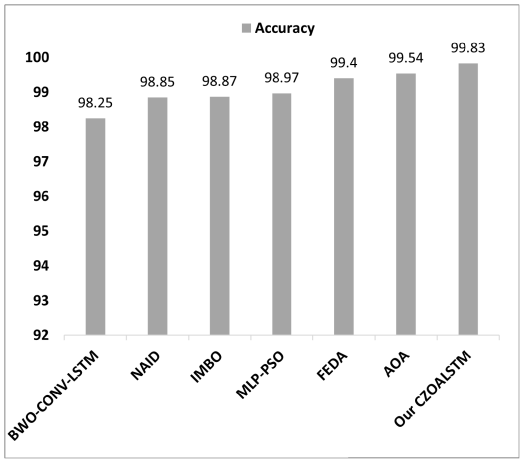
F. EFFICIENCY OF THE PROPOSED CZOLSTM FOR BINARY CLASSIFICATION

The purpose of this experiment is to evaluate the performance of the proposed CZOLSTM for binary classification. The binary classification classifies network traffic into two classes: 0 for normal and 1 for abnormal. This was done using the adjusted hyperparameters for the CZOLSTM, which are listed in Table 6. The pre-processed CSE-CICIDS2018 data set was used for this. To guarantee an equitable comparison, CZOLSTM, standard LSTM, and ZOA were compared and run in the same setting to demonstrate the efficiency of CZOLSTM for binary classification.

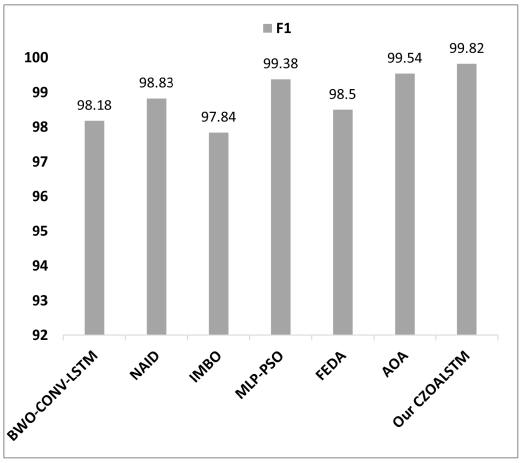
1) PERFORMANCE OF THE BINARY CLASSIFICATION OF THE STANDARD LSTM AND THE PROPOSED CZOLSTM

The performance of the proposed CZOLSTM is compared to that of the standard LSTM. The efficiency of the proposed CZOLSTM for binary classification in comparison to the standard LSTM is shown in Table 12. The CSE-CICIDS2018 dataset was utilized to obtain these results, which are the average of twenty runs, each with fifty epochs. This table demonstrates how effectively the proposed CZOLSTM algorithm is employed, with an accuracy of 99.83%, loss close to zero, an F1 score of 99.82, a precision of 99.83, a recall of up to 99.82, and an AUC of 99.82.

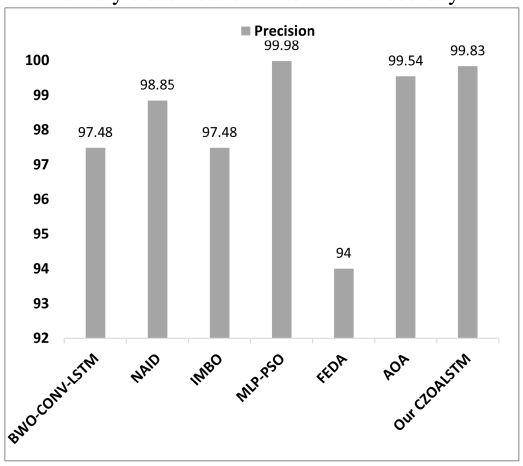
The best results in this table are indicated in bold text. Figures 8a, and 8b displayed the confusion matrix for a more thorough explanation of the experiment's results. Furthermore, the progress in accuracy among different runs is shown in Figure 4. Furthermore, the differences



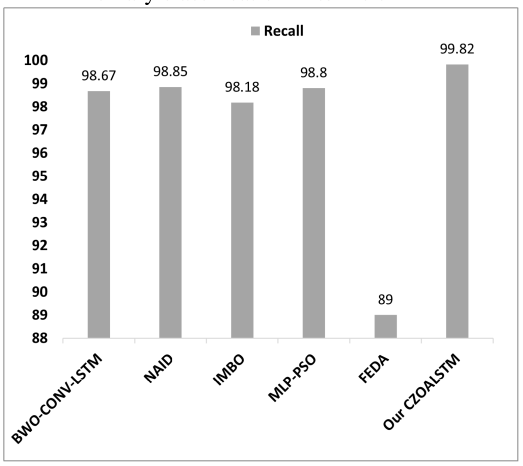
(a) Comparison with other algorithms based on binary classification in terms of Accuracy



(b) Comparison with other algorithms based on binary classification in terms of F1



(c) Comparison with other algorithms based on binary classification in terms of precision



(d) Comparison with other algorithms based on binary classification in terms of recall

FIGURE 7. **Comparison with other algorithms based on binary classification.**

between the CZOLSTM and the standard LSTM based on binary classification before and after FS are shown in Figure 6.

2) CZOLSTM VS. OTHER ALGORITHMS

To accurately evaluate the effectiveness of CZOLSTM, we conducted an in-depth comparison with the most recent DL algorithms that were employed in the literature on CTD. In this comparison, deep learning algorithms from previous studies such as [BWO-CONV-LSTM [38], NAID [63], IMBO [33], MLP-PSO [30], FEDA [36], and AOA [35]] were compared with our proposed CZOLSTM. A comparison between the CZOLSTM and other related algorithms based on binary classification is shown in Table 13. The bold text represents the overall best solution. The proposed CZOLSTM achieves an accuracy of 99.83, F1 of 99.82, precision of 99.83, recall of 99.82, loss close to zero, and AUC of 99.82. These results show that, in terms of the overall performance of binary classification, our proposed algorithm performs better than other algorithms. Figure 7 offers a clear explanation based on binary classification of how the proposed CZOLSTM outperformed other related literature in terms of accuracy, F1, precision, and recall.

G. EFFICIENCY OF THE PROPOSED CZOLSTM FOR MULTI-CLASS CLASSIFICATION

This experiment aims to multiclassify network traffic into 10 types of attacks (as listed in Table 5) against the benign class to assess the efficiency of the proposed CZOLSTM for cyber threat detection. The CZOA's adjusted hyperparameters, which are given in Table 6, were used for this. To guarantee an equitable comparison, the three algorithms - CZOLSTM, ZOA, and the standard LSTM - were implemented and run in the same setting. CZOLSTM's performance is evaluated by comparing it with the most recent literature by utilizing the CSE-CICIDS2018 dataset.

TABLE 14. Efficiency of the proposed CZOLSTM based on multi-class Classification.

Algorithm	Attack class	Accuracy	Loss	F1	Precision	Recall	AUC
CZOLSTM	FTP-BruteForce' 1	**99.99**	**0**	**100**	**100**	**100**	**100**
	SSH-Bruteforce 2	**99.99**	**0**	**100**	**100**	**100**	**100**
	DDOS attack-HOIC 3	**99.94**	**0**	**99.95**	**99.95**	**100**	**99.95**
	Bot 4	**99.92**	**0**	**99.96**	**99.96**	**100**	**99.96**
	DoS attacks-GoldenEye 5	**99.96**	**0**	**99.99**	**99.99**	**100**	**99.99**
	DoS attacks-Slowloris 6	**99.98**	**0**	**100**	**100**	**100**	**100**
	DDOS attack-LOIC-UDP 7	**100**	**0**	**100**	**100**	**100**	**100**
	Brute Force -Web' 8	**99.57**	**0.2**	**99.59**	**99.6**	**99.6**	**99.59**
	Brute Force -XSS 9	**99.88**	**0**	**99.95**	**99.96**	**100**	**99.95**
	SQL Injection' 10	**99.95**	**0**	**99.97**	**99.97**	**100**	**99.97**
ZOA	FTP-BruteForce' 1	99.95	0	99.92	99.92	99.92	99.92
	SSH-Bruteforce 2	99.96	0	99.95	99.95	99.95	99.95
	DDOS attack-HOIC 3	97.77	2.1	97.97	98.05	97.96	97.96
	Bot 4	97.68	2.1	97.56	97.65	97.6	97.6
	DoS attacks-GoldenEye 5	92.6	5	92.67	93.56	92.83	92.83
	DoS attacks-Slowloris 6	99.59	0.1	99.69	99.69	99.69	99.69
	DDOS attack-LOIC-UDP 7	100	0	100	100	100	100
	Brute Force -Web' 8	94.59	4.1	94.65	95.08	94.68	94.68
	Brute Force -XSS 9	94.59	3.3	94.68	94.83	94.68	94.68
	SQL Injection' 10	99.2	1.1	99.51	99.52	99.51	99.51
Standard LSTM	FTP-BruteForce' 1	99.95	1	99.97	99.97	99.97	99.97
	SSH-Bruteforce 2	98.24	2.3	99.38	99.39	99.38	99.38
	DDOS attack-HOIC 3	99.16	1.4	99.47	99.47	99.47	99.47
	Bot 4	95.68	4.5	96.6	96.73	96.6	96.6
	DoS attacks-GoldenEye 5	92.96	6.4	93.36	94.33	93.43	93.43
	DoS attacks-Slowloris 6	97.78	2.6	99.58	99.58	99.58	99.58
	DDOS attack-LOIC-UDP 7	99.99	1	100	100	100	100
	Brute Force -Web' 8	96.12	4.3	98.36	98.42	98.36	98.36
	Brute Force -XSS 9	96.78	3.4	96.86	97.02	96.87	96.87
	SQL Injection' 10	97.77	3.1	98.23	98.37	98.23	98.23

TABLE 15. Comparison between the CZOLSTM and Other Related Algorithms based on accuracy for multi-class classification.

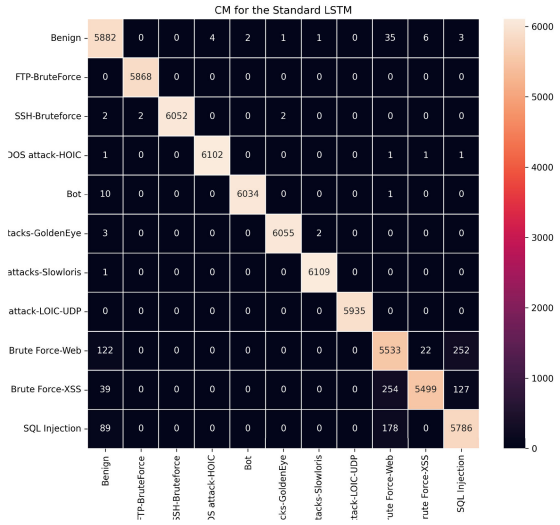
Study	Accuracy	Precision	F1	Recall
FEDA [36]	81	93	83	81
BWO-CONV-LSTM [38]	98.25	97.48	98.18	98.67
MLP-PSO [30]	98.41	99.55	99.2	98.83
IMBO [33]	98.87	97.48	97.84	98.8
AOA [35]	99.52	99.51	99.4	NA
Our Proposed CZOLSTM	**99.92**	**99.94**	**99.94**	**99.94**

1) PERFORMANCE OF THE MULTI-CLASS CLASSIFICATION OF THE STANDARD LSTM AND THE PROPOSED CZOLSTM

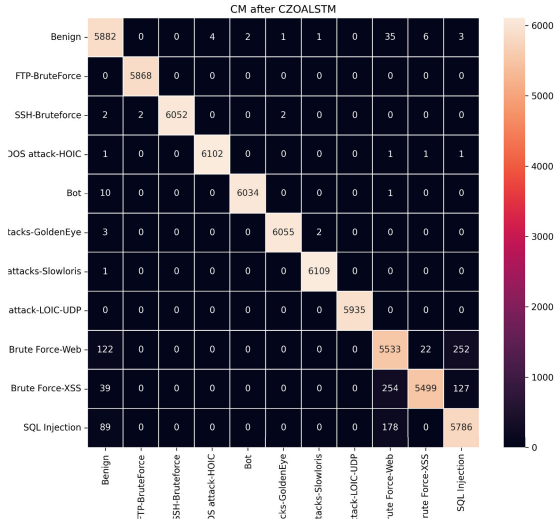
The purpose of this experiment is to use CSE-CIC-IDS2018 to investigate how well CZOLSTM correctly identifies ten different types of attack. For each attack type, accuracy, loss, F1 score, precision, recall, AUC, and CM were assessed. Additionally, a comparison was made between the outcomes of CZOLSTM and the standard LSTM.

To guarantee an equitable comparison, the three algorithms - CZOLSTM, ZOA, and the standard LSTM - were implemented and run in the same setting. The efficiency of the proposed CZOLSTM for multiclass classification in comparison to the standard LSTM and the ZOA algorithms is shown in Table 14. Furthermore, Figures 8a, and 8b display the confusion matrix based on multiclass classification for the standard LSTM and CZOLSTM respectively. Figures 10a, 10b, 10c, 10b, 10e, and 10f show the differences between CZOLSTM based on multi-class classification before and after employing FS in terms of Accuracy, Loss, F1, Precision, Recall, and AUC. It is evident from these outcomes that CZOLSTM outperformed other related algorithms across the board for each evaluation criteria. This is because the proposed CZOLSTM algorithm incorporates both CZOA for feature selection and manually optimized LSTM.

In multiclass classification, Table 14 demonstrates that the proposed CZOLSTM algorithm outperforms the standard



(a) CM for multi-class classification of the Standard LSTM (before FS)



(b) CM of CZOLSTM based on multi-class classification

FIGURE 8. CM of the Standard LSTM and CZOLSTM based on multi-class classification.

TABLE 16. Results of Wilcoxon Signed-Rank test.

Parameter	Value
P-value	0.03125
Surprisal (S-value)	5
Effect Size (r)	0.8793
Z	2.1539
W, (W-, W+)	0, (0, 21)
Non-zero difference pairs (n)	6
S.E	4.7697
Average of differences (xd)	47.42
SD of differences (Sd)	28.0897
Normality p-value	0.3006
Skewness	-0.9816

LSTM, with accuracy ranging from 99.57% to 100%. In contrast, the standard LSTM achieves less accuracy ranging from 92.96% to 99.95%. Furthermore, CZOLSTM

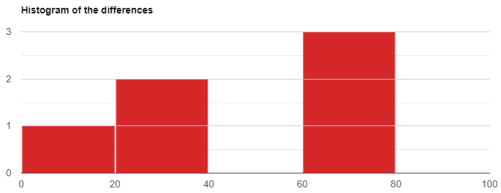


FIGURE 9. Histogram of the differences using the Wilcoxon Signed-Rank Test.

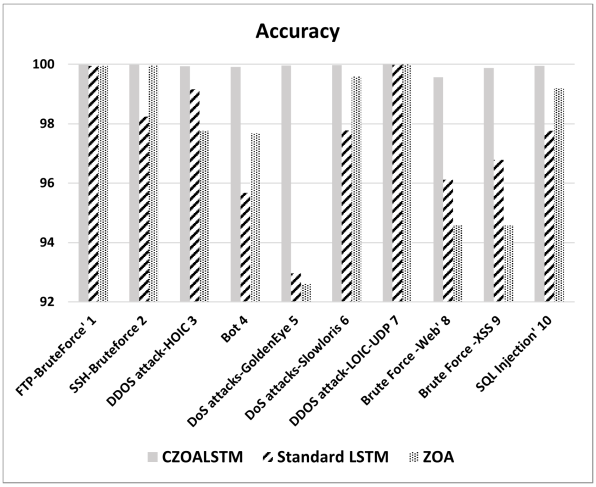
performs better than the ZOA algorithm, which achieves less accuracy levels between 92.6% and 100%. However, it's impressive that CZOLSTM performs the best performance for overall attacks.

Only 39 of the most important features were selected. After applying FS with the proposed CZOLSTM, the accuracy for the FTP-BruteForce attack was raised to 100, and the loss was zero. According to the SSH-Bruteforce attack, the accuracy reached 99.99, while the F1 score, precision, recall, and AUC attained 100. Referring to the DDOS-HOIC attack, the accuracy was 99.94, the loss was zero, and the F1 score, precision, recall, and AUC attained 99.95. For the bot attack, the accuracy was 99.92, the loss was zero, and the F1 score, precision, recall, and AUC were fixed at 99.96. Regarding the DoS-GoldenEye attack, accuracy reached 99.96, the loss was fixed at zero, the F1 score was 99.99, precision attained 99.99, and recall and AUC were all set to 100.

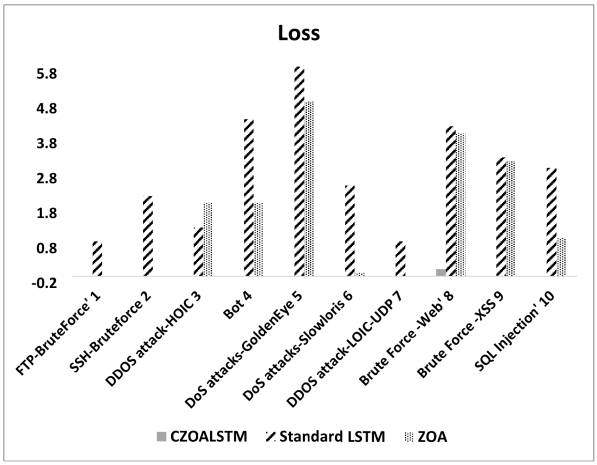
For the DoS-Slowloris attack, the accuracy was 99.98, and the loss was zero, while the F1 score, precision, recall, and AUC were all fixed at 100. On the other hand, for the DDOS-LOIC-UDP attack, the results were unexpected because accuracy, F1 score, precision, recall, and AUC were all fixed at 100. The loss was also zero. The Brute Force-Web attack achieved an accuracy of 99.57, a loss of 0.2, while the F1 score, precision, recall, and the AUC were fixed at 99.6. The accuracy of the Brute Force-XSS attack was 99.88, the loss was zero, the F1 score and AUC were 99.95, precision was 99.96, and recall reached 100. Last but not least, the SQL Injection attack achieved an accuracy of 99.95, a loss of zero, while the F1 score, precision, and AUC were all set to 99.97, and Recall was fixed at 100.

2) CZOLSTM VS. OTHER ALGORITHMS

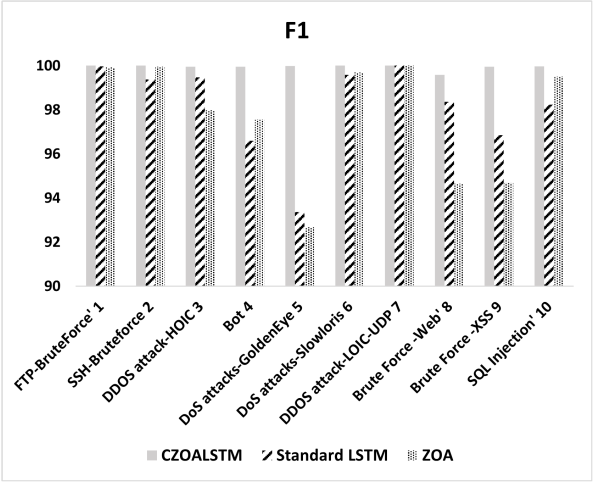
Similar to binary classification, we conducted an in-depth comparison with the most recent DL algorithms that were employed in the literature of CTD. In this comparison, deep learning algorithms from earlier studies such as [FEDA [36], BWO-CONV-LSTM [38], MLP-PSO [30], IMBO [33], and AOA [35]] were compared against our proposed CZOLSTM. The results of the comparison with other related algorithms are shown in Table 15. The bold text in the table represents the overall best solution. Figure 11 offers a clear explanation based on multi-class classification of how the proposed CZOLSTM outperformed other related literature in terms of accuracy, F1, precision, and recall.



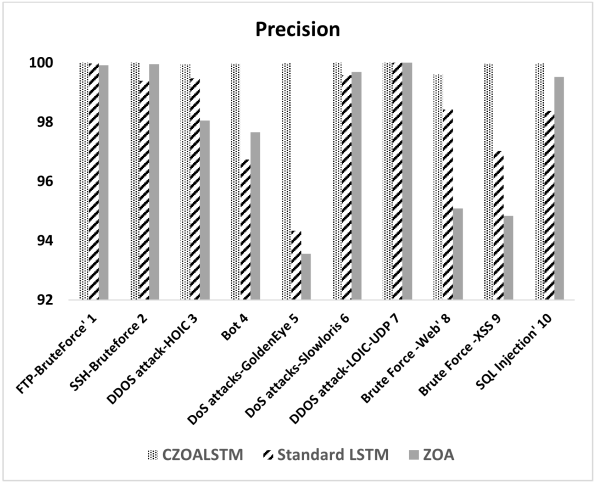
(a) Accuracy for CZOLSTM based on multiclass classification before and after FS



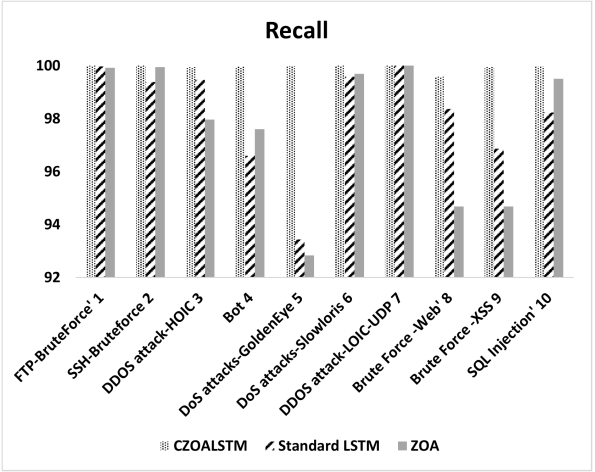
(b) Loss for CZOLSTM based on multiclass classification before and after FS



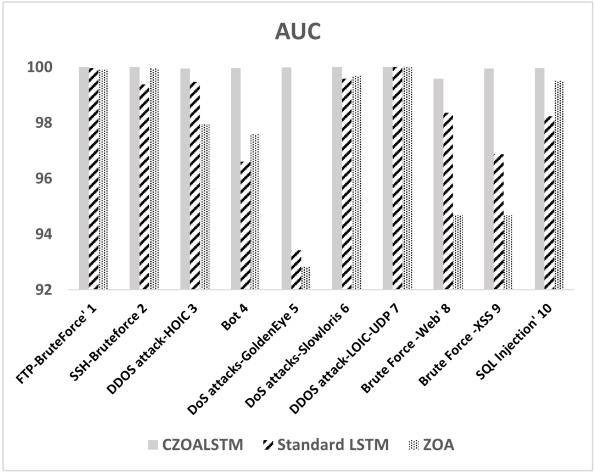
(c) F1 for CZOLSTM based on multiclass classification before and after FS



(d) Precision for CZOLSTM based on multiclass classification before and after FS



(e) Recall for CZOLSTM based on multiclass classification before and after FS



(f) AUC for CZOLSTM based on multiclass classification before and after FS

FIGURE 10. **Performance of multiclass classification before and after applying FS.**

For the FTP-Brute Force attack, CZOLSTM achieved an accuracy of 100 higher in multiclass classification than that reported in the other literature. Regarding the SSH-Bruteforce attack, our accuracy was 99.99 higher,

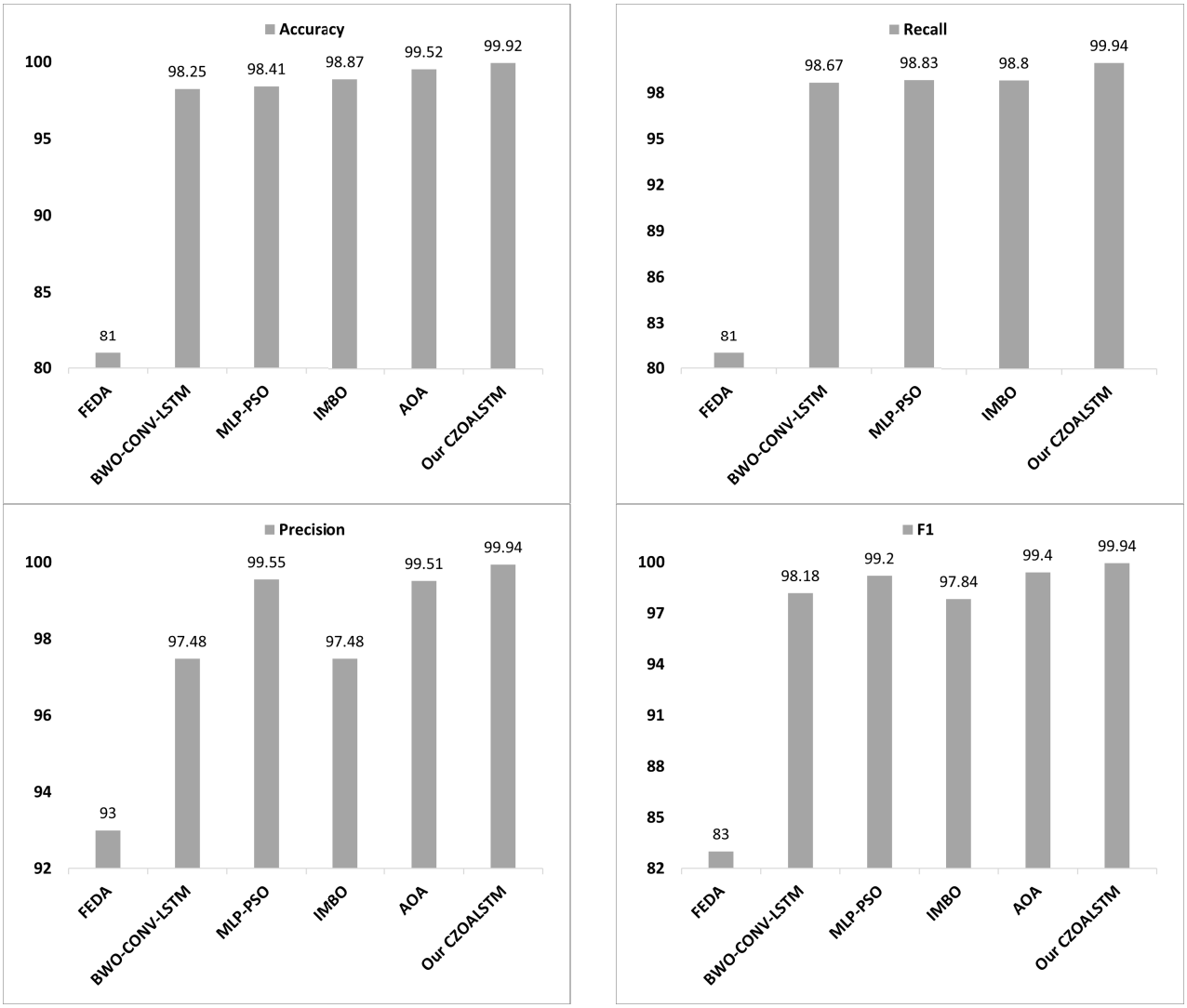


FIGURE 11. Comparsion between the proposed CZOLSTM based on multi-class classification with other related work.

respectively, than the other algorithms. For DDoS-HOIC and Bot attacks, Our obtained accuracy was 99.69 and 99.86 respectively, greater than that of BWO-CONV-LSTM [38]. According to the DoS-GoldenEye attack, the accuracy obtained was 99.95 higher than the accuracy obtained from [38] and [61]. Regarding the DoS-Slowloris attack, Our obtained accuracy was 99.98 higher than the other algorithms. According to Brute Force-Web, Brute Force-XSS, and SQL Injection attacks, our obtained accuracy reaches 99.86, 97.82, and 99.53 respectively which were the best results among other algorithms.

H. STATISTICAL ANALYSIS FOR THE PROPOSED ALGORITHM

In order to indicate the significant difference between the proposed algorithm and the other algorithms, we applied the Wilcoxon Signed-Rank test which is a non parametric test. It looks for a substantial difference between two independent groups in continuous or ordinal data. For our case study, we compared the results of the proposed CZOLSTM and the results obtained

The data from the two groups are combined in the test. The data is then sorted by value. The rank test compares the whole distributions, as opposed to the t-test which compares the averages of the groups. The test also compare the median of each group when the distributions of the two groups are comparable in form. Table 16 represents the results obtained from the Wilcoxon Signed-Rank test and Figure 9 shows the histogram of the differences using the Wilcoxon Signed-Rank Test.

we compared the values of the different evaluated performance metrics between the proposed CZOLSTM algorithm and the

V. CONCLUSION AND FUTURE WORK

Cyber threat detection systems have been used to detect malicious and abnormal network behavior. Different kinds of

these systems have been adopted to safeguard the network using a variety of DL algorithms. Optimizing system performance is the main objective of providing such consistent and effective systems. To address this problem, this paper has proposed a chaotic variant of the Zebra Optimization Algorithm (CZOLSTM) based on a hybrid Hénon and Gingerbread Chaotic Maps. The proposed algorithm invokes chaotic maps in ZOA to increase the diversity of the search process and to avoid trapping in the local minimum. It is well known that standard ZOA's performance is dependent on random numbers. However, if an appropriate zebra position is generated, this could lead to a good improvement in the obtained performance. Replacing the randomness of the optimization algorithm with a chaotic sequence improves overall performance. Consequently, the proposed improvements are used to select the optimal features for the CSE-CIC-IDS2018 dataset. The dataset contains the latest network attacks and employs LSTM to detect and classify cyber threats. In evaluating the effectiveness of the newly proposed CZOLSTM, binary and multi-class classifications are considered. The methodology's performance is compared against six cutting-edge algorithms for binary classification and five of them for multi-class classification. Other evaluation criteria such as accuracy, recall, F1 score, and precision have also been used for comparison. We concluded that the best performance would come from not utilizing all of the features in the dataset because some of them were redundant and unnecessary. The results showed that the best accuracy for binary classification is 99.83%, with F1-score of 99.82%, precision of 99.83%, and recall of 99.82%, in extensive and detailed experiments conducted on a real dataset. The best accuracy, F1-score, precision, and recall for multi-class classification were all around 99.92%, which gives the proposed algorithm an advantage over the compared ones. The acquired outcomes have been proven to demonstrate the efficiency of implemented improvements across many other algorithms.

The main purpose of the proposed CZOLSTM algorithm is to detect cyber threats, particularly intrusions, and to minimize or stop illegal access to any system, independent of the system's discipline. Any system need essentially be protected from unauthorized attacks, which is the primary goal of the proposed CZOLSTM algorithm. However, this does not imply that it cannot handle other technological challenges. Future work will examine the algorithm's effectiveness in a variety of contexts by taking multidisciplinary case studies into account and assessing CZOLSTM's performance using multipurpose datasets. In the future, the proposed algorithm will be expanded to include other public intrusion detection datasets. Also, it will be examined whether it is possible to further optimize performance through swarm optimization algorithms in order to detect attack classes with higher similarity, including low-frequency DDoS attacks. Furthermore, to enhance the IDS model, this variation in malicious traffic records will be investigated.

VI. DATA AVAILABILITY

Datasets related to this manuscript can be found at https://registry.opendata.aws/cse-cic-ids2018/, an open-source online data repository hosted at Canadian Institute for Cybersecurity [55].

REFERENCES

[1] A. K. Dey, G. P. Gupta, and S. P. Sahu, "A metaheuristic-based ensemble feature selection framework for cyber threat detection in IoT-enabled networks," *Decis. Anal. J.*, vol. 7, Jun. 2023, Art. no. 100206.

[2] W. A. H. M. Ghanem, S. A. A. Ghaleb, A. Jantan, A. B. Nasser, S. A. M. Saleh, A. Ngah, A. C. Alhadi, H. Arshad, A. H. Y. Saad, A. E. Omolara, Y. A. B. El-Ebiary, and O. I. Abiodun, "Cyber intrusion detection system based on a multiobjective binary bat algorithm for feature selection and enhanced bat algorithm for parameter optimization in neural networks," *IEEE Access*, vol. 10, pp. 76318–76339, 2022.

[3] M. A. Razib, D. Javeed, M. T. Khan, R. Alkanhel, and M. S. A. Muthanna, "Cyber threats detection in smart environments using SDN-enabled DNN-LSTM hybrid framework," *IEEE Access*, vol. 10, pp. 53015–53026, 2022.

[4] S. Yuan and X. Wu, "Deep learning for insider threat detection: Review, challenges and opportunities," *Comput. Secur.*, vol. 104, May 2021, Art. no. 102221.

[5] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proc. IEEE*, vol. 109, no. 3, pp. 247–278, Mar. 2021.

[6] A. Thakkar and R. Lohiya, "Role of swarm and evolutionary algorithms for intrusion detection system: A survey," *Swarm Evol. Comput.*, vol. 53, Mar. 2020, Art. no. 100631.

[7] D. Javeed, T. Gao, M. T. Khan, and D. Shoukat, "A hybrid intelligent framework to combat sophisticated threats in secure industries," *Sensors*, vol. 22, no. 4, p. 1582, Feb. 2022.

[8] S. Arora and P. Anand, "Binary butterfly optimization approaches for feature selection," *Expert Syst. Appl.*, vol. 116, pp. 147–160, Feb. 2019.

[9] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Comput. Elect. Eng.*, vol. 40, no. 1, pp. 16–28, Jan. 2014.

[10] A. Ibrahim, A. Ismail, H. Juahir, A. B. Iliyasu, B. T. Wailare, M. Mukhtar, and H. Aminu, "Water quality modelling using principal component analysis and artificial neural network," *Mar. Pollut. Bull.*, vol. 187, Feb. 2023, Art. no. 114493.

[11] X. Lei, Y. Xia, A. Wang, X. Jian, H. Zhong, and L. Sun, "Mutual information based anomaly detection of monitoring data with attention mechanism and residual learning," *Mech. Syst. Signal Process.*, vol. 182, Jan. 2023, Art. no. 109607.

[12] W. Shu, Z. Yan, J. Yu, and W. Qian, "Information gain-based semi-supervised feature selection for hybrid data," *Appl. Intell.*, vol. 53, no. 6, pp. 7310–7325, 2023.

[13] M. M. Mafarja, D. Eleyan, I. Jaber, A. Hammouri, and S. Mirjalili, "Binary dragonfly algorithm for feature selection," in *Proc. Int. Conf. New Trends Comput. Sci. (ICTCS)*, Oct. 2017, pp. 12–17.

[14] F. Wang, W. Zhang, Q. Yang, Y. Kang, Y. Fan, J. Wei, Z. Liu, S. Dai, H. Li, Z. Li, and L. Xu, "Generation of a Hutchinson–Gilford progeria syndrome monkey model by base editing," *Protein Cell*, vol. 11, no. 11, pp. 809–824, 2020.

[15] A. Yaqoob, N. K. Verma, and R. M. Aziz, "Optimizing gene selection and cancer classification with hybrid sine cosine and cuckoo search algorithm," *J. Med. Syst.*, vol. 48, no. 1, p. 10, Jan. 2024.

[16] R. M. Aziz, A. Hussain, and P. Sharma, "Cognizable crime rate prediction and analysis under Indian penal code using deep learning with novel optimization approach," *Multimedia Tools Appl.*, vol. 83, no. 8, pp. 22663–22700, Aug. 2023.

[17] I. Fister Jr., D. Fister, and X.-S. Yang, "A hybrid bat algorithm," 2013, *arXiv:1303.6310*.

[18] J. R. Lambert and E. Perumal, "Oppositional firefly optimization based optimal feature selection in chronic kidney disease classification using deep neural network," *J. Ambient Intell. Humanized Comput.*, vol. 13, no. 4, pp. 1799–1810, 2022.

[19] J. Nayak, H. Swapnarekha, B. Naik, G. Dhiman, and S. Vimal, "25 years of particle swarm optimization: Flourishing voyage of two decades," *Arch. Comput. Methods Eng.*, vol. 30, no. 3, pp. 1663–1725, Apr. 2023.

[20] A. Karaman, D. Karaboga, I. Pacal, B. Akay, A. Basturk, U. Nalbantoglu, S. Coskun, and O. Sahin, "Hyper-parameter optimization of deep learning architectures using artificial bee colony (ABC) algorithm for high performance real-time automatic colorectal cancer (CRC) polyp detection," *Appl. Intell.*, vol. 53, no. 12, pp. 15603–15620, 2023.

[21] A. Tiwari, "A hybrid feature selection method using an improved binary butterfly optimization algorithm and adaptive β–hill climbing," *IEEE Access*, vol. 11, pp. 93511–93537, 2023.

[22] S. Eskandari and M. Seifaddini, "Online and offline streaming feature selection methods with bat algorithm for redundancy analysis," *Pattern Recognit.*, vol. 133, Jan. 2023, Art. no. 109007.

[23] L. Sun, Y. Xin, T. Chen, and B. Feng, "Rolling bearing fault feature selection method based on a clustering hybrid binary cuckoo search," *Electronics*, vol. 12, no. 2, p. 459, Jan. 2023.

[24] P. Wang, B. Xue, J. Liang, and M. Zhang, "Feature selection using diversity-based multi-objective binary differential evolution," *Inf. Sci.*, vol. 626, pp. 586–606, May 2023.

[25] N. Bacanin, K. Venkatachalam, T. Bezdan, M. Zivkovic, and M. Abouhawwash, "A novel firefly algorithm approach for efficient feature selection with COVID-19 dataset," *Microprocessors Microsyst.*, vol. 98, Apr. 2023, Art. no. 104778.

[26] M. Ragab, "Hybrid firefly particle swarm optimisation algorithm for feature selection problems," *Expert Syst.*, vol. 41, no. 7, 2023, Art. no. e13363.

[27] Z.-M. Gao, J. Zhao, and Y.-J. Zhang, "Review of chaotic mapping enabled nature-inspired algorithms," *Math. Biosci. Eng.*, vol. 19, no. 8, pp. 8215–8258, 2022.

[28] A. Pourjabbar Kari, A. Habibizad Navin, A. M. Bidgoli, and M. Mirnia, "A new image encryption scheme based on hybrid chaotic maps," *Multimedia Tools Appl.*, vol. 80, pp. 2753–2772, Sep. 2021.

[29] R. Amin, G. El-Taweel, A. F. Ali, and M. Tahoun, "A hybrid extreme gradient boosting and long short-term memory algorithm for cyber threats detection," *MENDEL*, vol. 29, no. 2, pp. 307–322, Dec. 2023.

[30] S. Alzughaibi and S. El Khediri, "A cloud intrusion detection systems based on DNN using backpropagation and PSO on the CSE-CIC-IDS2018 dataset," *Appl. Sci.*, vol. 13, no. 4, p. 2276, Feb. 2023.

[31] Z. Liu, Y. Wang, F. Feng, Y. Liu, Z. Li, and Y. Shan, "A DDoS detection method based on feature engineering and machine learning in software-defined networks," *Sensors*, vol. 23, no. 13, p. 6176, Jul. 2023.

[32] A. A. E. Donkol, A. G. Hafez, A. I. Hussein, and M. M. Mabrook, "Optimization of intrusion detection using likely point PSO and enhanced LSTM-RNN hybrid technique in communication networks," *IEEE Access*, vol. 11, pp. 9469–9482, 2023.

[33] K. S. Babu and Y. N. Rao, "Improved monarchy butterfly optimization algorithm (IMBO): Intrusion detection using mapreduce framework based optimized ANU-Net," *Comput., Mater. Continua*, vol. 75, no. 3, p. 5887, 2023.

[34] R. Ghanbarzadeh, A. Hosseinalipour, and A. Ghaffari, "A novel network intrusion detection method based on metaheuristic optimisation algorithms," *J. Ambient Intell. Humanized Comput.*, vol. 14, no. 6, pp. 7575–7592, 2023.

[35] S. Fraihat, S. Makhadmeh, M. Awad, M. A. Al-Betar, and A. Al-Redhaei, "Intrusion detection system for large-scale IoT NetFlow networks using machine learning with modified arithmetic optimization algorithm," *Internet Things*, vol. 22, Jul. 2023, Art. no. 100819.

[36] S. Bhardwaj and M. Dave, "Enhanced neural network-based attack investigation framework for network forensics: Identification, detection, and analysis of the attack," *Comput. Secur.*, vol. 135, Dec. 2023, Art. no. 103521.

[37] Y. Zhang and Q. Liu, "On IoT intrusion detection based on data augmentation for enhancing learning on unbalanced samples," *Future Gener. Comput. Syst.*, vol. 133, pp. 213–227, Aug. 2022.

[38] P. R. Kanna and P. Santhi, "Hybrid intrusion detection using MapReduce based black widow optimized convolutional long short-term memory neural networks," *Expert Syst. Appl.*, vol. 194, May 2022, Art. no. 116545.

[39] E. Trojovská, M. Dehghani, and P. Trojovský, "Zebra optimization algorithm: A new bio-inspired optimization algorithm for solving optimization algorithm," *IEEE Access*, vol. 10, pp. 49445–49473, 2022.

[40] J. H. Notwell, "Exploring the genomic landscape of human disease," Ph.D. dissertation, Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, 2016.

[41] T. Caro, A. Izzo, R. C. Reiner, H. Walker, and T. Stankowich, "The function of zebra stripes," *Nature Commun.*, vol. 5, no. 1, p. 3535, Apr. 2014.

[42] J. Pastor, Y. Cohen, and N. T. Hobbs, "The roles of large herbivores in ecosystem nutrient cycles," *Conservation Biol. Cambridge*, vol. 11, p. 289, Nov. 2006.

[43] S. Arivazhagan, W. S. L. Jebarani, S. V. Kalyani, and A. Deiva Abinaya, "Mixed chaotic maps based encryption for high crypto secrecy," in *Proc. 4th Int. Conf. Signal Process., Commun. Netw. (ICSCN)*, Mar. 2017, pp. 1–6.

[44] N. Munir, M. Khan, A. Al Karim Haj Ismail, and I. Hussain, "Cryptanalysis and improvement of novel image encryption technique using hybrid method of discrete dynamical chaotic maps and Brownian motion," *Multimedia Tools Appl.*, vol. 81, no. 5, pp. 6571–6584, Feb. 2022.

[45] R. Nithya and D. Dhanasekaran, "Novel dominant color subband image encryption in visual sensor network for smart military surveillance system," *Traitement du Signal*, vol. 39, no. 3, pp. 951–960, 2022.

[46] L. Liu and J. Wang, "A cluster of 1D quadratic chaotic map and its applications in image encryption," *Math. Comput. Simul.*, vol. 204, pp. 89–114, Feb. 2023.

[47] T. S. Ali and R. Ali, "A novel medical image signcryption scheme using TLTS and Henon chaotic map," *IEEE Access*, vol. 8, pp. 71974–71992, 2020.

[48] C. Zhou and X. Chen, "Predicting China's energy consumption: Combining machine learning with three-layer decomposition approach," *Energy Rep.*, vol. 7, pp. 5086–5099, Nov. 2021.

[49] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[50] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Comput.*, vol. 12, no. 10, pp. 2451–2471, Oct. 2000.

[51] S. Ghimire, R. C. Deo, H. Wang, M. S. Al-Musaylh, D. Casillas-Pérez, and S. Salcedo-Sanz, "Stacked LSTM sequence-to-sequence autoencoder with feature selection for daily solar radiation prediction: A review and new modeling results," *Energies*, vol. 15, no. 3, p. 1061, Jan. 2022.

[52] P. K. Chawla, M. S. Nair, D. G. Malkhede, H. Y. Patil, S. K. Jindal, A. Chandra, and M. A. Gawas, "Parkinson's disease classification using nature inspired feature selection and recursive feature elimination," *Multimedia Tools Appl.*, vol. 83, pp. 35197–35220, Sep. 2023.

[53] A. I. Lawah, A. A. Ibrahim, S. Q. Salih, H. S. Alhadawi, and P. S. JosephNg, "Grey wolf optimizer and discrete chaotic map for substitution boxes design and optimization," *IEEE Access*, vol. 11, pp. 42416–42430, 2023.

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

[55] C. I. for Cybersecurity. (Jun. 2023). *A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018)*. [Online]. Available: https://registry.opendata.aws/cse-cic-ids2018/

[56] S. Alahmed, Q. Alasad, M. M. Hammood, J.-S. Yuan, and M. Alawad, "Mitigation of black-box attacks on intrusion detection systems-based ML," *Computers*, vol. 11, no. 7, p. 115, Jul. 2022.

[57] G. Mossa, G. Ghaleb, A. Faisal, A. Reem, and O. Suad, "A detailed analysis of benchmark datasets for network intrusion detection system," *Asian J. Res. Comput. Sci.*, vol. 7, no. 4, pp. 14–33, 2021.

[58] K. Jiyeon, S. Yulim, and C. Eunjung, "An intrusion detection model based on a convolutional neural network," *J. Multimedia Inf. Syst.*, vol. 6, no. 4, pp. 165–172, 2019.

[59] M. Mijwil, I. E. Salem, and M. M. Ismaeel, "The significance of machine learning and deep learning techniques in cybersecurity: A comprehensive review," *Iraqi J. Comput. Sci. Math.*, vol. 4, no. 1, pp. 87–101, Jan. 2023.

[60] R. K. Chouhan, M. Atulkar, and N. K. Nagwani, "A framework to detect DDoS attack in Ryu controller based software defined networks using feature extraction and classification," *Appl. Intell.*, vol. 53, pp. 4268–4288, Jun. 2022.

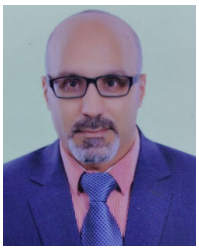
[61] M. Antunes, L. Oliveira, A. Seguro, J. Veríssimo, R. Salgado, and T. Murteira, "Benchmarking deep learning methods for behaviour-based network intrusion detection," *Informatics*, vol. 9, no. 1, p. 29, Mar. 2022.

[62] X. Ren, W. Yang, X. Jiang, G. Jin, and Y. Yu, "A deep learning framework for multimodal course recommendation based on LSTM+attention," *Sustainability*, vol. 14, no. 5, p. 2907, Mar. 2022.
[63] Y.-C. Wang, Y.-C. Houng, H.-X. Chen, and S.-M. Tseng, "Network anomaly intrusion detection based on deep learning approach," *Sensors*, vol. 23, no. 4, p. 2171, Feb. 2023.

REHAM AMIN received the B.Sc. and M.Sc. degrees in computer science from Suez Canal University, Ismailia, Egypt, in 2010, and 2017, respectively. She was a Ph.D. Student Intern with American University, Washington, DC, USA, in 2019, for six months. She was the Principal for systems and programs with the Information and Communication Technology Project, Suez Canal University. She is currently an Assistant Lecturer with the Faculty of Computers and Informatics, Suez Canal University. She has presented at conferences and published numerous papers in international journals and book chapters. Her research interests include machine learning, deep learning, swarm intelligence, biometric authentication, cyber security, and cloud computing.

GHADA EL-TAWEEL received the B.Sc. degree from Cairo University, in 1996, the M.Sc. degree from the Department of Computer Science, Helwan University, in 2000, and the Ph.D. degree from the Faculty of Computers and Information, Cairo University, in 2005. She was a Teaching Assistant with The American University in Cairo, from 1999 to 2004. From 2008 to 2010, she was an Executive Manager with the Quality Assurance Unit, Faculty of Computers and Informatics, Suez Canal University. Since October 2016, she has been a Professor with the Department of Computer Science. From September 2020 to July 2023, she was the Former Dean of the Faculty of Computers and Informatics, Suez Canal University. She is currently the Vice Dean of Education and Student Affairs with the Faculty of Computers and Informatics, Suez Canal University.

AHMED F. ALI received the B.Sc., M.Sc., and Ph.D. degrees in computer science from Assiut University, in 1998, 2006, and 2011, respectively. He was a Postdoctoral Fellow with Thompson Rivers University, Kamloops, BC, Canada, for one year. He was the Head of the Computer Science Department. He was the Vice Dean for Education and Student Affairs with the Egyptian Chinese College for Applied Technology, Suez Canal University, Ismailia, Egypt. He is currently a Professor with the Faculty of Computers and Informatics, Suez Canal University. He is also the Dean of the Faculty of Information Technology and Computer Science, Sinai University, Kantara Branch. His research has been focused on swarm intelligence and its applications, optimization, machine learning, data mining, web mining, and artificial intelligence. He has published many papers in international journals, book chapters, and conferences. He has many meta-heuristics lectures on the SlideShare website. He also served as a technical program committee member and a reviewer for worldwide conferences and international journals.



MOHAMED TAHOUN received the bachelor's degree in pure mathematics and computer science from the Faculty of Science, Menoufia University, Egypt, the M.Sc. degree in computer science from the Faculty of Computers and Information Science, Ain Shams University, Cairo, Egypt, in 2006, and the Ph.D. degree from the DAAD Channel Program between Humboldt University, Berlin, and Suez Canal University, Ismailia, Egypt, in 2015. He is currently an Associate Professor with the Computer Science Department, Faculty of Computers and Informatics, Suez Canal University. He has published research papers in the area of AI and computer vision applications. His research interests include medical image diagnosis, AI and machine learning approaches, the IoT and related applications, blockchain, and network applications.