AIMS Mathematics, 9(1): 22–37.
DOI: 10.3934/math.2024002
Received: 07 August 2023
Revised: 06 November 2023
Accepted: 13 November 2023
Published: 24 November 2023

http://www.aimspress.com/journal/Math

Research article

Inverse chi-square-based flamingo search optimization with machine learning-based security solution for Internet of Things edge devices

Youseef Alotaibi[1], R Deepa[2], K Shankar[3] and Surendran Rajendran[3,*]

[1] Department of Computer Science, College of Computer and Information Systems, Umm Al-Qura University, Makkah, 21955, Saudi Arabia
[2] Department of Computing Technologies, School of Computing, College of Engineering and Technology, SRM Institute of Science and Technology, SRM Nagar, Kattankulathur, Chennai, TN, India
[3] Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, 602105, India

* **Correspondence:** Email: surendran.phd.it@gmail.com; Tel: +919444013042.

Abstract: Internet of Things (IoT) edge devices are becoming extremely popular because of their ability to process data locally, conserve bandwidth, and reduce latency. However, with the developing count of IoT devices, threat detection, and security are becoming major concerns. IoT edge devices must avoid cyber threats and protect user data. These devices frequently take limited resources and can run on lightweight operating systems, which makes them vulnerable to security attacks. Intrusion detection systems (IDS) can be run on edge devices to recognize suspicious actions and possible risks. These systems monitor traffic patterns, and behavior, and identify attack signatures to detect and report on possible attacks. This study presents a design for an inverse chi square-based flamingo search optimization algorithm with machine learning (ICSFSO-ML) as a security solution for Internet of Things edge devices. The goal of the ICSFSO-ML technique is to apply ML and metaheuristics for threat recognition in IoT edge devices. To reduce the high dimensionality problem, the ICSFSO-ML technique uses the ICSFSO algorithm for feature selection purposes. Further, the ICSFSO-ML technique exploits the stacked bidirectional long short-term memory (SBiLSTM) model for the threat detection process. To enhance the efficacy of the SBiLSTM model, an arithmetic optimization algorithm (AOA) is applied for the hyperparameter selection process. The simulation performance of the ICSFSO-ML technique can be tested on a benchmark threat database. The performance analysis showed the benefits of the ICSFSO-ML methodology compared to existing methodologies with a

maximum accuracy of 98.22%.

Keywords: deep learning; Internet of Things; edge devices; machine learning; feature selection; security
Mathematics Subject Classification: 68M11, 68M25, 68T07, 68W1

1. Introduction

Recently, the Internet of Things (IoT) has experienced enormous growth in specific domain applications like smart transportation systems, industry, the medical field, and smart agriculture for increasing socio-economic development [1]. These IoT systems are formed by several interconnected actuators, various network-enabled devices, and sensors, which share several types of data through both private networks and internet platforms [2]. The Cisco investigation group forecast an average value of 75.3 billion effectively interconnected IoT devices by 2025. The absence of human intervention in data exchange among IoT systems makes it distinctive from standard internet technology [3]. The development of IoT devices has also improved the data network bandwidth requirements. However, many IoT devices have resource limitations, making it difficult to implement conventional security techniques for protecting systems against cyberattacks [4]. Major problems can occur with IoT devices that process sensitive data. Therefore, it is crucial to introduce mobile edge computing (MEC), which allows computation that is implemented at the network end to overcome resource-limit issues in IoT systems [5]. MEC permits IoT devices to offload more computationally intensive tasks to the proximal edge server. As the IoT develops as an industrial revolution, and systems collect live data, cybersecurity has become essential [6]. As a result, it is imperative to have a network intrusion detection system (NIDS), which could identify existing and forthcoming attacks to protect the IoT network and systems made on it.

Given the dependency of IoT methods on various edge devices and the part of the IoT in producing and collecting huge amounts of core data, accurate and effective techniques for identifying anomalous behavior in IoT edge devices are required [7]. Machine learning (ML) has been employed from the centralized cloud. However, a structure that requires a lot of resources can be placed so that it can consistently have as much processing power, storage space, and power as it requires to process data. However, there might be further waiting time because of network delays from the device to the cloud and back, as well as the quantity of data in applications with higher traffic. This increases expenses concerning delay and financial expenditure. The basic changes are required in AI approaches and cloud-to-device intention to provide effectual, maintainable, and acceptable solutions for the predicted potential demands [8]. Hence, it is essential to manage the increasing processing needs and traffic, and minimize delays [9]. The major problem preventing edge devices from performing at their maximum potential is the lack of resources among these limited resources, and edge devices are needed by AI applications. Generally, edge devices are very small in their physical sizes, with low power capacity and processing ability [10]. Security solutions for IoT edge devices have several applications to safeguard a connected ecosystem. These solutions can be applied in various fields such as healthcare, critical infrastructure, and maintaining public safety; in industrial settings, to safeguard manufacturing processes and prevent unauthorized access to machinery; and in home automation, where they protect personal information and ensure the security of smart appliances and devices.

This study presents a design for an inverse chi square-based flamingo search optimization algorithm with machine learning (ICSFSO-ML) as a security solution for Internet of Things edge devices. The goal of the ICSFSO-ML technique is to apply ML and metaheuristics for threat recognition in IoT edge devices. To reduce the high dimensionality problem, the ICSFSO-ML technique uses the ICSFSO algorithm for feature selection. Further, the ICSFSO-ML technique exploits the stacked bidirectional long short-term memory (SBiLSTM) model for the threat detection process. To enhance the efficacy of the SBiLSTM model, an arithmetic optimization algorithm (AOA) is applied for the hyperparameter selection process. The simulation performance of the ICSFSO-ML technique is tested using a benchmark threat database.

2. Related works

Mishra et al. [11] aimed to detect potential attacks on various types of networks. The IoT anomalies are identified by inspiring message queuing telemetry transport (MQTT) across a virtual network. Dey et al. [12] developed a metaheuristic-based intelligent structure for cyberattack identification employing ensemble FS and a classification model. Initially, a metaheuristic-based ensemble FS method was developed utilizing binary grey wolf optimization (BGWO) and binary gravitational search algorithm (BGSA). Then, RF and AdaBoost can be utilized for detecting and classifying cyberattacks. In [13], the authors studied effective attack identification approaches for these software-defined IoT (SD-IoT) networks. Then, the impacts of RF and ML techniques in different feature groups (for example IPs and ports) could be analyzed for the identification accuracy of various attacks.

Aldaej et al. [14] examined current security and privacy issues affecting a network of drones (NoD). A hybrid ML approach of LR and RF was employed with the aim of classification of data samples for maximum efficiency. By integrating complex AI-inspired methods into the architecture, the presented approach alleviated cybersecurity vulnerabilities while creating a protected and secured NoD. In [15], the authors suggested a collaborative cyberattack intelligence-sharing method for allowing several organizations to combine forces in the framework, estimation, and training of a robust ML-based network IDS. This technique employs two main features for its application: the accessibility of network data traffic in a standard design and implementation of a federated learning method to prevent the need to share sensitive user data among organizations. Haddad Pajouh et al. [16] recommended a protected design for the IoT edge layer structure named AI4SAFE-IoT. The modules developed in the research included cyberattack allocation, intelligent web application firewall, cyberattack hunting, and cyberattack intelligence.

Mozo et al. [17] introduced an analysis of the incorporation of ML modules in a distributed scenario. A real-time developing attack vector (crypto-mining malware attack) was employed as a demonstration. The overall potential of recent green AI methods was integrated for optimizing the size and complexity of ML approaches to decrease their energy consumption while retaining their capability for accurately detecting possible cyberattacks. Gasu [18] presented an attentive literature study of ML and data mining (DM) techniques for cyber analytics in aid of cyberattack identification and intrusion detection.

3. The proposed model

This manuscript provides an automated ICSFSO-ML-based security solution for IoT edge devices.

The major aim of the ICSFSO-ML technique is to apply ML and metaheuristics for threat recognition in IoT edge devices. In the proposed ICSFSO-ML algorithm, three main phases are contained: ICS-FSO-based feature selection, SBiLSTM-based detection, and AOA-based parameter tuning. Figure 1 depicts the entire flow of the ICSFSO-ML methodology.

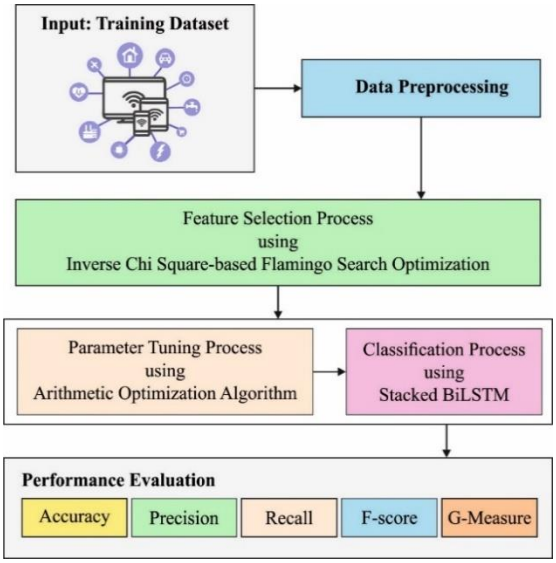

Figure 1. The overall flow of the ICSFSO-ML algorithm.

3.1. Feature selection using ICS-FSO algorithm

Primarily, the ICSFSO-ML technique uses the ICSFSO algorithm for feature selection purposes. The ICS-FSO model controls the global searching space in the foraging range of flamingos to maintain a balance between the exploration and exploitation stages [19]. Moreover, the current chi-square test in FSO becomes locked for the independent variable, which leads to a high error rate. To overcome these challenges, the working mechanism of ICS is paired with FSO. The proposed model takes fitness value as a better feature score and flamingo as the feature. The following steps provide a detailed description of ICS-FSO:

Step 1: At first, the population of the flamingos was initialized in an attempt to generate the optimal performance dependent upon the data available, and then the search region was chosen, while the food accessibility was rich. Consider the Flamingo (Γ_j) having an abundance of food in the j^{th} parameter.

Consider Γ_{ij} to be the coordinates of the i^{th} and j^{th} flamingo population parameters; in this case, the flamingos' inhabitants must contend with the problems of sporadic access to food and inaccurate data transmission. This error can be estimated by the maximum distance of the flamingo's beak scan from the foraging behavior, as shown below:

$$\left|\Im_1 \times \Gamma_j^b + \Im_2 \times \Gamma_{ij}^t\right| \tag{1}$$

In Eq (1), $\wp_2$ denotes a randomly generated value within $[-1,1]$, and $\Im_1$ refers to a figure that was unintentional to exploit the uniform distribution. In beak behavior, the range of scanning can be maintained in the range as,

$$\Im_2 \times \left|\Im_1 \times \Gamma_j^b + \wp_2 \times \Gamma_{ij}^t\right|. \tag{2}$$

Considering Γ_j^b as food for a large population, the flamingo distances are modified, and the traveling is calculated by $\wp_1 \times \Gamma_j^b$, where $\wp_1$ refers to a random integer within $[-1,1]$. Lastly, the foraging movement of the flamingo from the t^{th} step can be estimated by the flamingo beak scan range, and the distance among the moving feet is shown as follows:

$$d_{ij}^t = \wp_1 \times \Gamma_j^{bi} + \Im_2 \times \left|\Im_1 \times \Gamma_j^{bi} + \Im_2 \times \Gamma_{ij}^t\right|. \tag{3}$$

The location of flamingo's foraging is represented as,

$$\Gamma_{ij}^{t+1} = \frac{\left(\Gamma_{ij}^t + \wp_1 \times \Gamma_j^{bt} + \wp_2 \times \left|\wp_1 \times \Gamma_j^{bt} + \wp_2 \times \Gamma_{ij}^t\right|\right)}{K}. \tag{4}$$

In Eq (4), Γ_{ij}^t shows the location of the i^{th} flamingo at the j^{th} variable in the ith iteration of the flamingo's population, Γ_j^{bt} indicates the j^{th} flamingo size with the better fitness from the population at t iteration, Γ_{ij}^{t+1} denotes the location of the i^{th} flamingo at the j^{th} population parameter from the $(t+1)th$ iteration,

$$K\left(\frac{1}{\Gamma}, \eta\right) = \left[\frac{2^{-\eta/2}}{\gamma(\eta/2)}\Gamma^{-\eta/2-1}e^{-\frac{1}{2\Gamma}}\right]. \tag{5}$$

In Eq (5), $K(\eta)$ denotes the diffusion factor that follows in the inverse chi-square distribution of η degrees of freedom, and γ represents the gamma function. The foraging range dimension has been improved, and the simulation can be done for the individual chosen, it boosted the capacity for meritocracy all over the globe. $\Im_1 = N(0,1)$ and $\Im_2 = N(0,1)$ a random integer generated using a uniform distribution, $\wp_1$ and $\wp_2$ are altered in [-1,1].

Step 2: The flamingos have migrated to the following region because of the food scarcity from the existing region. Given the fact that the region in question exhibits a significant level of dietary consumption j^{th} variable is Ab_j, the migration of the flamingo population can be given below:

$$\Gamma_{ij}^{t+1} = \Gamma_{ij}^t + \omega \times \left(\Gamma_j^{bt} - \Gamma_{ij}^t\right). \tag{6}$$

In Eq (6), $\omega = N(0,n)$ indicates the randomly generated value based on the uniform distribution with n degrees of freedom, which increases the searching space and arbitrariness behavior of the specific flamingo with a particular migration method employed for inspiration. At last, the maximal iteration attained gives an optimum performance and value that can be replaced from the main function and arranged to evaluate the essential feature, and the relevant features can be framed inside the data frame employing ICS-FSO:

$$\Omega_1 = A\left[\Gamma_1^d, \Gamma_2^d, \Gamma_3^d, \Gamma_4^d, \Gamma_5^d, \Gamma_6^d, \Gamma_7^d \ldots \Gamma_n^d\right]. \tag{7}$$

3.2. Detection using SBiLSTM model

At this stage, the SBiLSTM model is used for the threat detection process. LSTM is distinct from

the RNN technique, which is generally deployed to process time sequence data [20]. It can be established to resolve the issue of RNNs' effort in long-term learning and dependencies. The main idea of LSTM exists from its memory cell with a gated function. Its gated method contains 3 gates: forget (f_t), input (i_t) and output gates (o_t). An input gate (i_t) controls the input of present data. Once the input data comes into the unit, an input gate carries out a computation to determine whether to input the present data. The memory gate (f_t) controls the maintenance of past data. Once the past data comes into the unit, the memory gate executes a computation to determine how to maintain the data. The output gate (O_t) manages the outcome of the present data. It is defined as outputting the present data by carrying out a computation. Furthermore, C_t signifies the long-term memory unit, whereas h_t signifies the short-term memory unit.

$$\mathrm{f_t} = \sigma(\mathrm{w_f} \cdot [\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_f}). \tag{8}$$

$$\mathrm{i_t} = \sigma(\mathrm{w_i} \cdot [\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_i}). \tag{9}$$

$$\mathrm{C'_t} = \tanh(\mathrm{w_c} \cdot [\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_c}). \tag{10}$$

$$\mathrm{O_t} = \sigma(\mathrm{w_o} \cdot [\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_0}). \tag{11}$$

$$\mathrm{C_t} = \mathrm{f_t} \cdot \mathrm{C_{t-1}} + \mathrm{i_t} \cdot \mathrm{C'_t}. \tag{12}$$

$$\mathrm{h_t} = \mathrm{O_t}.\tanh(\mathrm{C_t}). \tag{13}$$

In the above equations, σ denotes the sigmoid activation function. The variables w and b in the equations represent the weighted and intercepted, correspondingly. BILSTM is a distinct LSTM that integrates a further layer of reverse computation with the base LSTM. The original series is $(\mathrm{A_0}, \mathrm{A_1}, \mathrm{A_2}, \ldots, \mathrm{A_i})$, but the reversed series is defined as $(\mathrm{A'_0}, \mathrm{A'_1}, \mathrm{A'_2}, \ldots, \mathrm{A_i})$. The last resultant value is defined by the forward as well as reverse sequences:

$$\mathrm{y_i} = \mathrm{v_1} \cdot \mathrm{A_i} + \mathrm{v_2} \cdot \mathrm{A'_i}. \tag{14}$$

In this equation, $\mathrm{v_1}$ and $\mathrm{v_2}$ represent the equivalent weights linked with 2 sequences.

An SBiLSTM has an NN structure that contains stacking multi-layers of BiLSTMs on top of one another. Figure 2 depicts the framework of stacked BiLSTM. The outcome of the 1st BiLSTM layer serves as the input to the 2nd BiLSTM layer, etc. All the subsequent layers capture a high-level representation of input data, potentially contributing to a more expressive and greater model for sequence processing tasks.

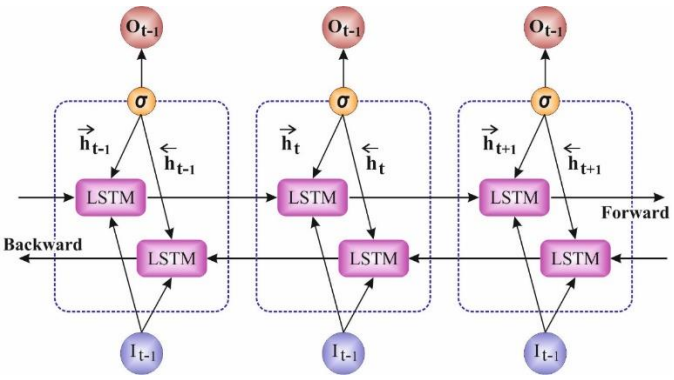

Figure 2. The architecture of Stacked BiLTSM.

3.3. AOA-based hyperparameter tuning

The hyperparameters related to the SBiLSTM approach are elected by the AOA. AOA is based on the searching process for the potential region from the candidate solution based on subtraction, division, multiplication and addition operators [21]. The efficiency of the optimization approach is measured with two search processes called exploitation and exploration. The process conducts the local search in the exploitation phase, concentrating on promising regions to search for a better solution. During the exploration phase, the process performs a global search to prevent local solutions. In this work, division $(\div)$ and multiplication $(\times)$ take the larger steps from the searching space and are thereby employed for exploratory search. Addition $(+)$ and subtraction $(-)$ operators can find capable areas by taking smaller steps from the searching space, which provides the best exploitation search. The AOA begins with an arbitrary candidate solution (X). A better solution should be found through exploration and exploitation. Among the X_n candidate solutions. MOA is used to define whether to implement an exploitation or exploration search as follows:

$$\mathrm{MOA}(t_{\mathrm{CURRENT}}) = \mathrm{MOA}_{\min} + t_{\mathrm{CURRENT}} \times \left(\frac{\mathrm{MOA}_{\max} - \mathrm{MOA}_{\min}}{t_{\mathrm{MAX}}}\right). \tag{15}$$

In Eq (15), t_{CURRENT} indicates the existing iteration ranges from 1 to r_{MAX} (maximal iteration), and $\mathrm{MOA}_{\max}$, and $\mathrm{MOA}_{\min}$ determine the minimal and maximal values of MOA. The AOA algorithm utilizes the 3 randomly generated values (rand_1, rand_2 and rand_3) between zero and one. Once an arbitrary number rand_1 is bigger than MAO, AOA enters the exploration stage. Now, if $\mathrm{rand}_2 > 0.5$, the place of the s^{th} solution can be upgraded by the division operators. Else, if rand_2 is lesser than 0.5, then the AOA model exploits the multiplication operator for updating the position of the s^{th} outcome.

$$S_{s,l}(t_{\mathrm{CURRENT}} + 1) = \begin{cases} \mathrm{Condition1: best}(S,) \div (\mathrm{MOP}+\in) \times [(\mathrm{UB}_l - \mathrm{LB}_l) \times \mu + \mathrm{LB}_l], \mathrm{rand}_2 < 0, \\ \mathrm{Condition2: best}(S,) \times \mathrm{MOP} \times [(\mathrm{UB}_l - \mathrm{LB}_l) \times \mu + \mathrm{LB}_l], \mathrm{otherwise}, \end{cases} \tag{16}$$

where $S_{s,l}(t_{\mathrm{CURRENT}} + 1)$ denotes the s-th solution in the l^{th} location, and best_{x_1} shows the best performance from the l^{th} place. ε indicates the integer with a smaller value. and μ represents the control parameter characterized by predetermined or fixed values of 0.5 for updating the exploration search. The upper and lower boundaries are characterized as the UB_l and LB_l parameters.

$$\mathrm{MOP}(t_{\mathrm{CURRENT}}) = 1 - \frac{t_{\mathrm{CURRENT}}}{t_{\mathrm{Max}^{1/\alpha}}} 1/\alpha. \tag{17}$$

The coefficients in the context of math optimizer probability (MOP) serve a specific purpose and are explicitly defined in the t^{th} iteration. The AOA model enters into the exploitation stage if $\mathrm{rand}_1 <$ MOA function. Moreover, the model updates the position of the s-th solution using subtraction. If $\mathrm{rand}_3 > 0.5$. Otherwise, the place of the s^{th} solution can be upgraded by addition $(+)$ operator as follows:

$$S_{s,l}(f_{\mathrm{CURRENT}} + 1) = \begin{cases} \mathrm{Condition1: best}(S_l) \div (\mathrm{M0P}+\in) \times [(\mathrm{UB}_l - \mathrm{LB}_l) \times \mu + \mathrm{LB}_l], \mathrm{rand}_2 < 0, \\ \mathrm{Condition2: best}(S_l) \times \mathrm{M0P} \times [(\mathrm{UB}_l - \mathrm{LB}_l) \times \mu + \mathrm{LB}_l], \mathrm{otherwise}. \end{cases} \tag{18}$$

Algorithm 1: Pseudocode of AOA.

Initialize the parameters α, μ, $\mathrm{Max}_{\mathrm{iter}}$, $\mathrm{MOA}_{\mathrm{max}}$, $\mathrm{MOA}_{\mathrm{min}}$
Initialize the position of each solution randomly as
$$s = 1, \ldots, n$$
while $(t_{CURRENT} < t_{Max})$ do
Compute the Fitness Function (FF) $(\mathrm{Fitness}_{\mathrm{F}})$ to provide a solution
Define the best solution
Upgrade the value of MOA based on Eq (15).
Upgrade the value of MOP based on Eq (17).
for ($s = 1$ to Solution) do
for ($l = 1$ to Location) do
Produce an arbitrary value within $[0,1](\mathrm{rand}_1, \mathrm{rand}_2, \text{ and } \mathrm{rand}_3)$
if $\mathrm{rand}_1 > \mathrm{MOA}$ then
Exploration stage
if $\mathrm{rand}_2 > 0.5$ then
Apply the Division operator
Upgrade position l of s^{th} solution based on condition1 in Eq (17)
else
Apply the Multiplication operator
Upgrade position l of s^{th} solution based on condition2 in Eq (16)
end if
else
Exploitation stage
if $\mathrm{rand}_3 > 0.5$ then
Apply the Subtraction operator.
Upgrade place l of s^{th} solution based on condition1 in Eq (18)
else
Apply the Addition operator
Upgrade position l of s^{th} solution based on condition2 in Eq (18)
end if
end if
end for
end for
$t_{CURRENT} = t_{CURRENT} + 1$
end while
Return the best solution (S_{best}).

Fitness choice is a key feature of the AOA methodology. An encoded outcome was deployed to assess better candidate performances. Presently, the accuracy value is the major condition deployed to design an FF.

$$\mathrm{Fitness} = \max(P), \tag{19}$$

$$P = \frac{TP}{TP + FP}, \tag{20}$$

where FP and TP denote the false and true positive values.

4. Result and discussion

The performance of the ICSFSO-ML technique was tested on the anomaly database [22,23]. The database has 20000 instances and 2 class labels as represented in Table 1. Among the available 13 features, the ICSFSO algorithm has chosen 8 features.

Table 1. Description of the dataset.

Class	No. of Samples
Normal Status	10000
Anomalies	10000
Total Samples	20000

The anomaly detection results of the ICSFSO-ML technique are depicted in Figure 3. The confusion matrices demonstrate the effectual recognition of the normal and anomalous samples under all classes [24–26].

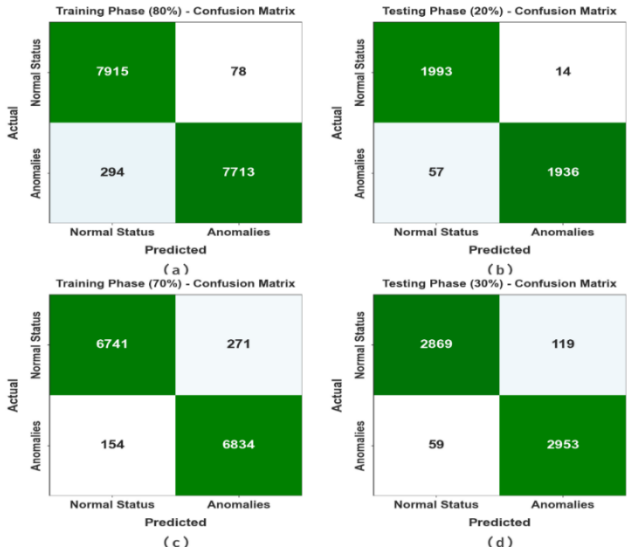

Figure 3. Confusion matrices of (a-b) 80:20 of TR set/TS set and (c-d) 70:30 of TR set/TS set.

The anomaly identification results of the ICSFSO-ML technique are tested with 80:20 of TR set/TS set as shown in Table 2 and Figure 4. The outcome showed that the ICSFSO-ML system recognized the normal and anomaly classes. On 80% of the TR set, the ICSFSO-ML algorithm offers average accuracy, prec_n, recall, $F-\text{score}$ and G_{measure} of 97.68%, 97.71%, 97.68%, 97.67% and 97.68%, respectively. Also, on 20% of the TS set, the ICSFSO-ML method achieves average accuracy, prec_n, recall, $F-score$, and G_{measure} of 98.22%, 98.25%, 98.22%, 98.22% and 98.23%, correspondingly.

Table 2. Anomaly identification outcome of ICSFSO-ML technique on 80:20 of TR set/TS.

Class	$Accu_y$	$Prec_n$	$Reca_l$	F_{Score}	$G_{Measure}$
TR set (80%)					
Normal Status	99.02	96.42	99.02	97.70	97.71
Anomalies	96.33	99.00	96.33	97.65	97.65
Average	97.68	97.71	97.68	97.67	97.68
TS set (20%)					
Normal Status	99.30	97.22	99.30	98.25	98.26
Anomalies	97.14	99.28	97.14	98.20	98.21
Average	98.22	98.25	98.22	98.22	98.23

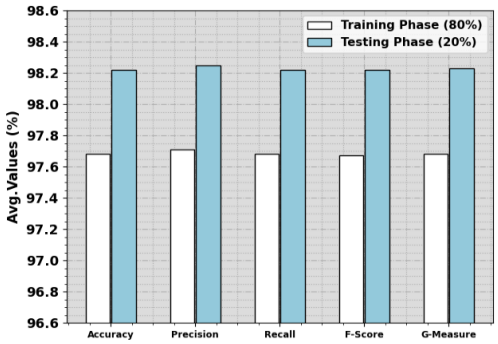

Figure 4. Average outcome of ICSFSO-ML technique on 80:20 of TR set/TS set.

The anomaly identification outcome of the ICSFSO-ML methodology was tested with 70:30 of the TR set/TS setting as portrayed in Table 3 and Figure 5. The simulation values depicted that the ICSFSO-ML algorithm recognized the normal and anomaly classes. On 70% of the TR set, the ICSFSO-ML system had average accu_y, prec_n, reca_l, F_score, and G_measure of 96.97%, 96.98%, 96.97%, 96.96% and 96.97% correspondingly. Then, on 20% of the TS set, the ICSFSO-ML methodology achieved average accu_y, prec_n, reca_l, F_score, and G_measure of 97.03%, 97.06%, 97.03%, 97.03% and 97.04%, respectively.

Table 3. Anomalies identification outcome of ICSFSO-ML technique on 70:30 of TR set/TS.

Class	$Accu_y$	$Prec_n$	$Reca_l$	F_{Score}	$G_{Measure}$
TR set (70%)					
Normal Status	96.14	97.77	96.14	96.94	96.95
Anomalies	97.80	96.19	97.80	96.98	96.99
Average	96.97	96.98	96.97	96.96	96.97
TS set (30%)					
Normal Status	96.02	97.98	96.02	96.99	97.00
Anomalies	98.04	96.13	98.04	97.07	97.08
Average	97.03	97.06	97.03	97.03	97.04

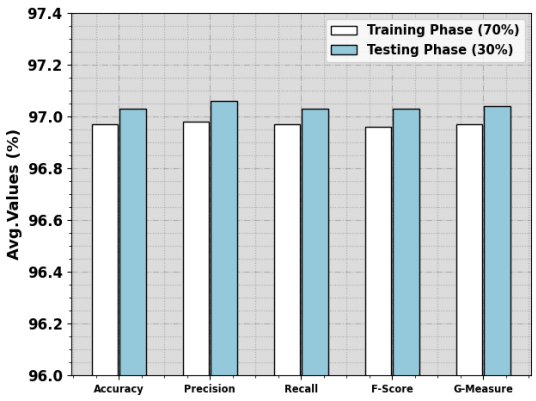

Figure 5. The average outcome of the ICSFSO-ML technique on 70:30 of the TR set/TS set.

Figure 6 illustrates the training accuracy TR_accu_y and VL_accu_y of the ICSFSO-ML methodology on 80:20 of the TR set/TS set. The TL_accu_y was determined by the evaluation of the ICSFSO-ML method on the TR dataset, whereas the VL_accu_y was computed by evaluating the performance on a separate testing dataset. The outcome exhibits that TR_accu_y and VL_accu_y upsurge with a rise in epochs. Therefore, the performance of the ICSFSO-ML method increased on the TR and TS datasets with an upsurge in several epochs.

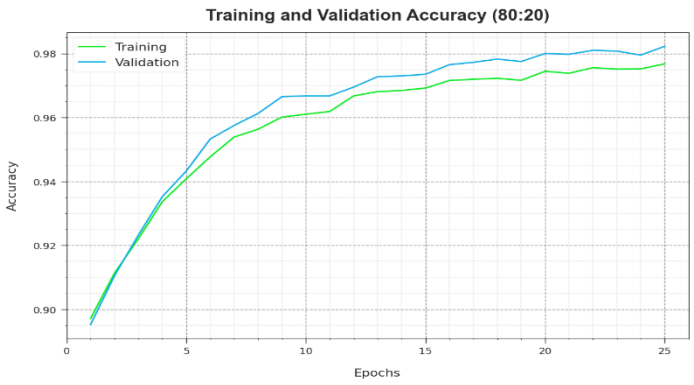

Figure 6. Accu_y curve of ICSFSO-ML technique on 80:20 of TR set/TS set.

In Figure 7, the TR_loss and VR_loss results of the ICSFSO-ML approach on 80:20 of the TR set/TS set are shown. The TR_loss defines the error among the predictive outcome and original values on the TR data. The VR_loss signifies the measure of the performance of the ICSFSO-ML technique on individual validation data. The outcomes point out that the TR_loss and VR_loss tend to reduce with rising epochs, portraying the higher performance of the ICSFSO-ML system and its ability to produce an accurate classification. The lesser values of TR_loss and VR_loss establish the improved performance of the ICSFSO-ML methodology in capturing patterns and relationships.

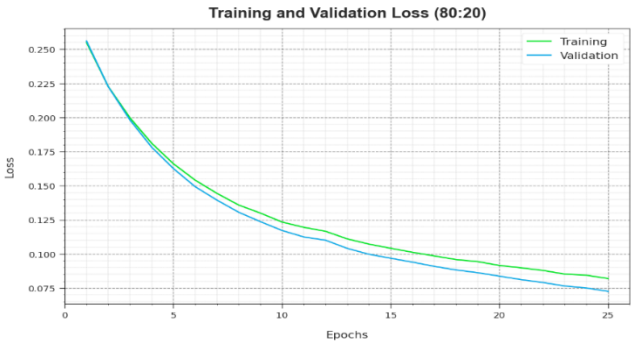

Figure 7. Loss curve of ICSFSO-ML technique on 80:20 of TR set/TS set.

In Figure 8, a comprehensive precision-recall (PR) investigation of the ICSFSO-ML approach is shown for 80:20 of the TR set/TS set in Figure 8. The simulation values demonstrated that the ICSFSO-ML system led to enhanced PR outcomes. Further, it was obvious that the ICSFSO-ML method has greater PR outcomes in the 2 classes.

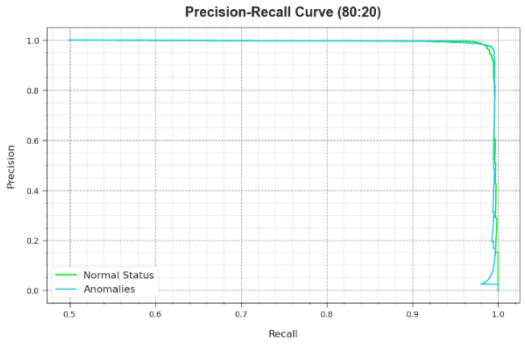

Figure 8. PR curve of ICSFSO-ML technique on 80:20 of TR set/TS set.

In Figure 9, an ROC analysis of the ICSFSO-ML system is demonstrated on 80:20 of the TR set/TS set. The simulation results show that the ICSFSO-ML approach led to greater values of ROC. Also, it was apparent that the ICSFSO-ML methodology achieved better ROC outcomes in the 2 classes.

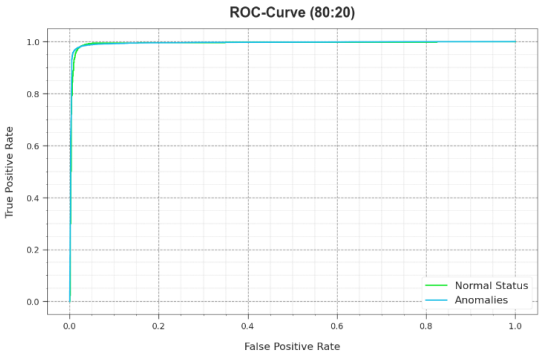

Figure 9. PR curve of ICSFSO-ML technique on 80:20 of TR set/TS set.

Table 4 and Figure 10 illustrate the comparison outcomes of the ICSFSO-ML approach with other models in terms of different metrics [27–29]. The table values portrayed the ineffectual performance of the AE model, whereas the SVM and Anomaly-IDS models have shown somewhat better results over the AE model. Along with that, the DEA and LR approaches have exhibited close results. However, the DT and RF systems have accomplished considerable results, the ICSFSO-ML technique ensured better performance with maximum $prec_n$, $accu_y$, F_{score} and $reca_l$ of 98.25%, 98.22%, 98.22% and 98.22%, respectively. These outcomes show the better performance of the ICSFSO-ML algorithm.

Table 4. Comparative outcome of ICSFSO-ML technique with other approaches.

Model	$Prec_n$	$Accu_y$	F_{Score}	$Reca_l$
ICSFSO-ML	98.25	98.22	98.22	98.22
DT Model	95.40	96.70	92.10	93.30
LR Model	93.10	94.50	91.70	92.50
SVM Model	92.60	95.30	90.00	91.00
RF Model	95.50	97.20	93.20	94.40
Deep Autoencoder	93.79	94.71	92.71	93.46
Autoencoders	89.81	84.86	90.85	91.90
Anomaly-IDS	92.65	91.65	93.16	92.24

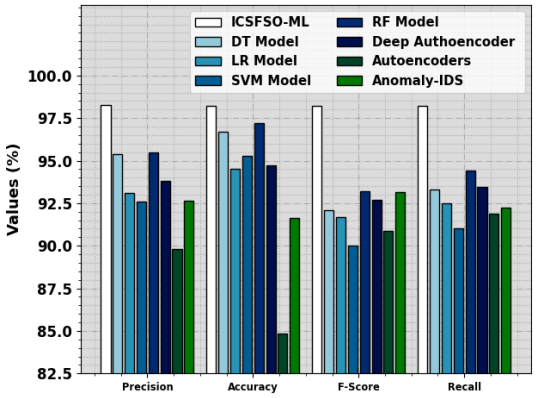

Figure 10. Comparative outcome of ICSFSO-ML technique with other approaches.

5. Conclusions

This manuscript has provided an automated ICSFSO-ML-based security solution for IoT edge devices. The major aim of the ICSFSO-ML technique is to apply ML and metaheuristics for threat recognition in IoT edge devices. In the proposed ICSFSO-ML algorithm, three major phases are contained: ICS-FSO-based feature selection, SBiLSTM-based detection and AOA-based parameter tuning. Primarily, the ICSFSO-ML approach uses the ICSFSO algorithm for feature selection purposes, which reduces the computation complexity and boosts the classification results. In addition, the ICSFSO-ML technique makes use of the SBiLSTM model for the threat detection process. To enhance the efficacy of the SBiLSTM model, AOA is applied for the hyperparameter selection process. The simulation value of the ICSFSO-ML technique is made on the benchmark threat database. The performance showed the benefits of the ICSFSO-ML approach compared to existing methods with a

maximum accuracy of 98.25%. In the future, the ICSFSO-ML technique holds significant potential for further advancements and applications in the field of IoT security. Additional research could focus on refining and optimizing the technique to address emerging IoT security threats and vulnerabilities. This might involve extending the methodology to accommodate a broader range of IoT devices and communication protocols. Furthermore, integrating real-time monitoring and response capabilities, as well as considering the scalability of the solution for large-scale IoT deployments, could be valuable avenues of exploration.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Funding

This research has been funded by the Deanship for Research & Innovation, Ministry of Education in Saudi Arabia, through project number: IFP22UQU4281768DSR205.

Acknowledgments

The authors extend their appreciation to the Deanship for Research & Innovation, Ministry of Education in Saudi Arabia, through project number: IFP22UQU4281768DSR205.

Conflict of interest

The authors declare no conflict of interest.

References

1. A. Mozo, A. Karamchandani, L. de la Cal, S. Gomez-Canaval, A. Pastor, L. Gifre, A mchine-learning-based cyberattack detector for a cloud-based SDN controller, *Apli. Sci.*, **13** (2023), 4914. https://doi.org/10.3390/app13084914
2. A. Dutta, S. Kant, Implementation of cyber threat intelligence platform on the Internet of Things (IoT) using TinyML approach for deceiving cyber invasion, *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, 2021, 1–6. https://doi.org/10.1109/ICECCME52200.2021.9590959
3. A. Aldaej, T. A. Ahanger, M. Atiquzzaman, I. Ullah, M. Yousufudin, Smart cybersecurity framework for IoT-empowered drones: Machine learning perspective, *Sensors*, **22** (2022), 2630. https://doi.org/10.3390/s22072630
4. I. Goni, J. M. Gumpy, T. U. Maigari, M. Muhammad, A. Saidu, Cybersecurity and cyber forensics: Machine learning approach, *Mach Learn Res.*, **5** (2020), 46–50. https://doi.org/10.11648/j.mlr.20200504.11

5. F. S. Alrayes, N. Alshuqayran, M. K. Nour, M. Al Duhayyim, A. Mohamed, A. A. A. Mohammed, et al., Optimal fuzzy logic enabled intrusion detection for secure IoT-cloud environment, CMC-*Comput. Mater. Con.*, **74** (2023), 6737–6753. http://doi.org/10.32604/cmc.2023.032591
6. P. Koloveas, T. Chantzios, S. Alevizopoulou, S. Skiadopoulos, C. Tryfonopoulos, Intime: A machine learning-based framework for gathering and leveraging web data to cyber-threat intelligence, *Electronics*, **10** (2021), 818. https://doi.org/10.3390/electronics10070818
7. M. Maray, H. M. Alshahrani, K. A. Alissa, N. Alotaibi, A. Gaddah, A. Meree, Optimal deep learning driven intrusion detection in SDN-Enabled IoT environment, *Comput. Mater. Con.*, **74** (2023), 6587–6604. https://doi.org/10.32604/cmc.2023.034176
8. K. H. Almotairi, Application of internet of things in the healthcare domain, *J. Umm Al-Qura Univ. Eng. Architecture*, **14** (2023), 1–12. https://doi.org/10.1007/s43995-022-00008-8
9. T. Moulahi, R. Jabbar, A. Alabdulatif, S. Abbas, S. El Khediri, S. Zidi, et al., Privacy-preserving federated learning cyber-threat detection for intelligent transport systems with blockchain-based security, *Expert Syst.*, **40** (2023), 13103. https://doi.org/10.1111/exsy.13103
10. K. Marsh, S. E. Gharghasheh, Fuzzy Bayesian learning for cyber threat hunting in industrial control systems, In: *Handbook of big data analytics and forensics*, Springer, Cham. 2022, 117–130. https://doi.org/10.1007/978-3-030-74753-4_8
11. S. Mishra, A. Albarakati, S. K. Sharma, Cyber threat intelligence for IoT using machine learning, *Processes*, **10** (2022), 2673. https://doi.org/10.3390/pr10122673
12. A. K. Dey, G. P. Gupta, S. P. Sahu, A metaheuristic-based ensemble feature selection framework for cyber threat detection in IoT-enabled networks, *Decis. Anal. J.*, **7 (**2023), 100206. https://doi.org/10.1016/j.dajour.2023.100206
13. Y. Zhang, J. Xu, Z. Wang, R. Geng, K. K. R. Choo, J. A. Perez-Díaz, et al., Efficient and intelligent attack detection in software-defined IoT networks. In: *2020 IEEE International Conference on Embedded Software and Systems (ICESS)*, 2020. https://doi.org/10.1109/ICESS49830.2020.9301591
14. A. Aldaej, T. A. Ahanger, M. Atiquzzaman, I. Ullah, M. Yousufudin, Smart cybersecurity framework for IoT-empowered drones: Machine learning perspective, *Sensors*, **22** (2022), 2630. https://doi.org/10.3390/s22072630
15. M. Sarhan, S. Layeghy, N. Moustafa, M. Portmann, Cyber threat intelligence sharing scheme based on federated learning for network intrusion detection, *J. Netw. Syst. Manage.*, **31**(2023), 3. https://doi.org/10.1007/s10922-022-09691-3
16. H. HaddadPajouh, R. Khayami, A. Dehghantanha, K. K. R. Choo, R. M. Parizi, AI4SAFE-IoT: An AI-powered secure architecture for edge layer of the Internet of things, *Neural Comput. Appl.*, **32** (2020), 16119–16133. https://doi.org/10.1007/s00521-020-04772-3
17. H. Makina, A. B. Letaifa, Bringing intelligence to Edge/Fog in Internet of Things-based healthcare applications: Machine learning/deep learning-based use cases, *Int. J. Commun. Syst.*, **36** (2023), e5484. https://doi.org/10.1002/dac.5484
18. D. K. Gasu, Threat detection in cyber security using data mining and machine learning Techniques, In: *Modern theories and practices for cyber ethics and security compliance*, IGI Global, 2020, 234–253.
19. M. Dahiya, N. Nitin, Developing a secure framework using feature selection and attack detection technique, *Comput. Mater. Con.*, **74** (2023), 4183–4201. https://doi.org/10.32604/cmc.2023.032430

20. K. S. Riya, R. Surendran, C. A. T. Romero, M. S. Sendil, Encryption with user authentication model for internet of medical things environment, *Intell. Autom. Soft Comput.*, **35** (2023), 507–520. https://doi.org/10.32604/iasc.2023.027779
21. N. Talpur, S. J. Abdulkadir, E. A. P. Akhir, M. H. Hasan, H. Alhussian, M. H. A. Abdullah, A novel bitwise arithmetic optimization algorithm for the rule base optimization of the deep neuro-fuzzy system, *J. King Saud Univ.-Com.*, **35** (2023), 821–842. https://doi.org/10.1016/j.jksuci.2023.01.020
22. K. Nagappan, S. Rajendran, Y. Alotaibi, Trust aware multi-objective metaheuristic optimization based secure route planning technique for cluster-based IoT environment, *IEEE Access*, **10** (2022), 112686–112694. https://doi.org/10.1109/ACCESS.2022.3211971
23. A. Yazdinejad, B. Zolfaghari, A. Dehghantanha, H. Karimipour, G. Srivastava, R. M. Parizi, Accurate threat hunting in industrial internet of things edge devices, *Digit. Commun. Netw.*, **9** (2023), 1123–1130. https://doi.org/10.1016/j.dcan.2022.09.010
24. R. Surendran, Y. Alotaibi, A. Subahi, Lens-oppositional wild geese optimization based clustering scheme for wireless sensor networks assists real time disaster management, *Comput. Syst. Sci. Eng.*, **46** (2023), 835–851. https://doi.org/10.32604/csse.2023.036757
25. M. O. Pahl, F. X. Aubet, All eyes on you: Distributed multi-Dimensional IoT microservice anomaly detection, In: *2018 14th International Conference on Network and Service Management (CNSM)*, 2018, 72–80.
26. X. Yang, S. Li, Prediction of COVID-19 using a WOA-BILSTM model, *Bioengineering*, **10** (2023), 883. https://doi.org/10.3390/bioengineering10080883
27. F. Gabbay, R. L. Aharoni, O. Schweitzer, Deep neural network memory performance and throughput modeling and simulation framework, *Mathematics*, **10** (2022), 4144. https://doi.org/10.3390/math10214144
28. J. Mariselvam, S. Rajendran, Y. Alotaibi, Reinforcement learning-based AI assistant and VR play therapy game for children with Down syndrome bound to wheelchairs, *AIMS Mathematics*, **8** (2023), 16989–17011. https://doi.org/10.3934/math.2023867
29. T. Tamilvizhi, Y. Alotaibi, S. Rajendran, K. Nagappan, Improved wolf swarm optimization with deep-learning-based movement analysis and self-regulated human activity recognition, *AIMS Mathematics*, **8** (2023), 12520–12539. https://doi.org/10.3934/math.2023629

AIMS Press