A hybrid attack detection strategy for cybersecurity using moth elephant herding optimisation-based stacked autoencoder

Abdullah Shawan Alotaibi

Department of Computer Science, Shaqra University, Shaqra, Kingdom of Saudi Arabia

Correspondence
Abdullah Shawan Alotaibi, Department of Computer Science, Shaqra University, Shaqra, Kingdom of Saudi Arabia.
Email: a.shawan@su.edu.sa

Abstract
Cybersecurity is a major concern in the network resources of the Internet of Things (IoT) environment, as the attacks on the IoT devices degrade the performance of the computational operations. Various attack detection methods are adopted in the research area to prevent an illegal user from accessing the resources. To resolve the vulnerabilities in the computing devices, an effective attack detection method, named, Moth Elephant Herding Optimisation (MEHO)-based stacked autoencoder approach is proposed in this research. Initially, the input data is passed into the pre-processing stage, where the data is cleaned by removing the noise and artefacts. The pre-processed data is further subjected to the feature selection stage, where the Class-Wise Information Gain technique (CIG) is used to select the essential features. The class-aware features, like traffic-based features, content-based features and basic features are selected efficiently. Finally, the attack detection is performed using the stacked autoencoder classifier, which is trained using the proposed MEHO algorithm. The MEHO algorithm is developed by integrating the Moth search (MS) algorithm and the Elephant Herding Optimisation (EHO). The performance is evaluated using the metrics, like accuracy, False Acceptance Rate (FAR) and detection rate, acquired with the values of 0.9286, 0.0636 and 0.9258 respectively.

1 | INTRODUCTION

The increasing technology in the Internet of Things (IoT) environment bridges the information gap between the physical background and the IoT devices by integrating the miniaturised sensing, remote control devices and the sophisticated devices using the processing capability, internet-based communications and the storage facilities. The results revealed in most of the mission applications, like healthcare [1, 2], infrastructure management, emergency, environmental monitoring [3], defence, image processing [4] and traffic, indicate that the intelligence and the business science of the materials are allowed to enhance the human interface with the computing services. The research works are recently focussing to migrate the integrated large-scale IoT technology to the cloud platform by allowing the physical devices, such as actuators, sensors and controllers to be visualised in the form of cloud resources, hence, these functions can be accessed on-demand through various tenants [5]. The major and the most feasible architecture [6] used to control and monitor the smart devices, like Smart Buildings (SB) and Smart Homes (SH) are the Supervisory Control and Data Acquisition (SCADA) system and Building Automation Systems (BAS). As the SCADA and the BAS systems are interconnected through the Internet services and the network resources, these systems are considered as the primary target for the cyber adversaries, because it is not purposely designed to resolve the cyber threats [7]. If the IoT devices are allowed to be used in an isolated environment, then, failures and vulnerabilities may arise due to the careless policies and misconfigurations, which result in inaccurate computation analysis and loss of control in the IoT environment [8].

Cyberattack captures the psychological pressure and the password of the user to violate the person’s information [9]. Cybersecurity is mainly used to focus on the computing systems for processing their information and to exchange the data in the respective channels; the violation which arises will be notified and is endorsed under the criminal law [10]. Intrusion Detection System (IDS) is one of the security mechanisms used to detect the cyberattack in the network. In the IoT network, various smart objects are interconnected, which range from the supercomputer to the tiny...
devices, as it has less computation power, hence providing security to these devices is a challenging task and so the cybersecurity is a major loophole of the IoT network implementation [11]. IDS is an essential hardware or the software security tool used to eliminate the threats which otherwise prevents the abuse or illegal resource accesses and reports the attacks to the responsible provider, who provides the security [12]. At first, the attack detection was launched in the computer security threat surveillance and monitoring system. The major reason for using the IDS is to detect the attacks which are not prevented through other security measures [9].

These security solutions are defined using the Q-learning mechanism which is based on the vulnerability analysis methods [13]. Different cybersecurity approaches are introduced in the big data security, such as privacy protection using a virtual ring framework [14], authorisation and authentication approach [15], analysis using the taxonomy framework [16] and Merkle tree-based handshaking approach [17]. In Ref. [18], an automated security mechanism is developed to provide cybersecurity in the network system. The deep learning approach uses the autoencoder to extract the features and Deep Belief Networks (DBN) to perform the attack detection and is also applicable for detecting the codes in the malicious devices. The statistical methods, like Gaussian Mixture Distribution, Chi-square distribution and principal component analysis, are used to detect the intrusion from the devices. The intrusions are effectively detected using the methods like rule-based fuzzy logic, support vector machine and artificial neural network [9].

The main intention of this research is to design and develop an effective attack detection algorithm for providing cybersecurity in the IoT devices. The proposed Moth Elephant Herding Optimisation (MEHO) algorithm involves three stages to detect the cyberattacks, which include, pre-processing, feature selection and attack detection stage. At first, the input data is pre-processed to enhance the quality of data by removing the noise and artefacts from the data. The pre-processed data is further employed in the feature selection phase, where the suitable features are effectively selected using the Class-Wise Information Gain technique (CIG) technique. The selected features are processed by the attack detection phase using the proposed MEHO-based stacked autoencoder approach which effectively detects the cyberattacks in the IoT devices.

The motivation of using CIG is to measure the goodness of a feature for recognising a specific class, and it also helps to select the features with the highest information content for a specific class. The most important advantage is that it is used to detect both malware loaders and infected executables. It has a high information gain.

The contribution of the work is as follows:

- The class-aware features, like traffic-based features, content-based features and basic features are selected using the CIG technique. CIG recognises more useful features based on the available class information.
- The stacked autoencoder classifier detects the cyberattacks using the proposed MEHO algorithm based on the selected features.

The rest of this article is organised as: the literature review using the existing techniques along with their merits and demerits which are discussed in Section 2. The proposed attack detection method is elaborated in Section 3, and the results along with the performance analysis are elaborated in Section 4. Finally, the conclusion is made in Section 5.

2 | LITERATURE SURVEY

Various literature reviews are surveyed as: in 2017, Kim et al. [19] introduced a cybersecurity enhancement model, which was used to offer reliable services in the IoT environment. The errors were easily predicted from the physical situations and the cyber operations are effectively performed. In this method, the capacity of the security infrastructure was strengthened, but the blocking difficulties and the path backtracking were not addressed. Further in 2018, Kozik et al. [5] modelled an extreme learning machine model to perform traffic classification which was based on the flexibility of cloud-based architectures integrated with large-scale machine learning approaches. However, it failed to face the security problems of the cyber-physical system. In 2018, Jiang et al. [20] developed an attack detection approach using the multi-channel scheme to solve the security problem in the network. Here the detection rate, channel detection and feature abstractions were integrated with the detection framework. The attacks were detected from normal traffic by using the voting algorithm. This approach efficiently detects the attacks and thus attained better accuracy. However, the performance of this mechanism was poor, for the new types of attacks. In 2018, Azmoodeh et al. [21] introduced the deep eigenspace-based learning approach to detect internet malware using the operational code sequence of the device. The benign and the malicious applications were easily classified using the learning mechanism. This approach was more robust in detecting the junk code attacks, but it failed to deploy the IoT nodes in the IoT system. In 2018, Rathore and Park [22] modelled the Fuzzy c-means algorithm to classify the data as either normal or attack. This algorithm solved the labelled data problems and offered better generalisation performance. Even though it attained a better accuracy rate, the processing speed was very low. In 2018, Dido and Chilamkurti [23] introduced a deep learning-based attack detection approach to detect the cyberattacks in the network. It used the sharing parameters and eliminates the local minima from the network. However, this method failed to detect the intrusions by considering the payload data. In 2019, Sani et al. [12] developed an identity-based security approach to manage the energy in the smart grid framework. This approach was more efficient and secure in the energy internet. However, it failed to include the broadcast and the multi-cast communication in the energy internet. In 2019, Aldaej [24] introduced an intrusion prevention approach to increase cybersecurity in the IoT environment. It assures the survivability,
security and network performance during the incidence of attack. Anyhow, the resource and the accessibility of the service were decreased, while dealing with huge nodes. Jasiul et al. [25], developed a cyberthreats detection model, based on the coloured Petri nets. The detection process was done by the digital signature matching and the analysis of the behavioural models. This method was used to prevent and detect the advanced persistent threats. Anyhow, it was impossible to use the virtualised machines in this method. Bojovic et al. [26], developed a hybrid two-fold model based on the exponential moving average algorithm, which was used to detect the distributed denial-of-service (DDoS) attacks. This method offered better results in recall, detection rate, F1 score and precision. Anyhow, this method did not differentiate between peer-to-peer traffic and denial-of-service traffic. Catak and Mustacoglu [27], developed a method based on deep learning which was used to classify the network traffics. The main advantage of this method was the detection of DDoS. Anyhow, varying the time could affect the classification results of the network traffic.

3 | PROPOSED MOTH-BASED ELEPHANT HERDING OPTIMISATION ALGORITHM FOR ATTACK DETECTION

In this research work, the attack detection strategy to provide cybersecurity in the IoT environment is performed by developing a MEHO algorithm. The proposed MEHO algorithm involves three stages to detect the cyberattacks which include, pre-processing, feature selection and attack detection stage. Initially, the input data is passed into the pre-processing stage, where the data is cleaned to eliminate the noise and artefacts. Hence, the pre-processed data offers a high-quality data which are fed into the feature selection stage. The feature selection process is carried out in the feature selection stage to select the essential features to perform the attack detection. The class-aware features, like traffic-based features, content-based features and basic features are selected using the CIG technique. Accordingly, the selected features are subjected to the attack detection phase for detecting the attacks. The attack detection is performed using the proposed MEHO-based stacked autoencoder approach, where the stacked autoencoder classifier is used to classify whether the traffic is an attack or not. The stacked autoencoder classifier is trained using the newly designed hybrid optimisation algorithm named MEHO, which is the integration of the Moth search (MS) [28] algorithm and the Elephant Herding Optimisation (EHO) [29–32] for attack detection in IoT. Figure 1 shows the block diagram of the proposed MEHO-based stacked autoencoder approach.

3.1 | Pre-processing

Initially, the input data is subjected to the pre-processing stage, where the missing data is effectively imputed. Moreover, the processing enhanced the effectiveness of the input data, which is further used to perform the feature selection process.

3.2 | Feature selection using class-wise information gain technique

Once the data is pre-processed, it is further subjected to the feature selection stage, where the class-aware features are effectively selected. It is required to perform the feature selection process to reduce the complexity of the framework and make it easier to interpret. Furthermore, selecting accurate features helps to reduce the overfitting and to enhance the accuracy of the model. The features using CIG are selected based on the statistical information of the features. The major advantage of using CIG is to recognise the appropriate features is based on available class information. The selected features are categorised into three different types as, basic features, content-based features and traffic-based features. The class-aware features are selected using the CIG technique which overcomes the imperfection of the global features and plans to identify more useful features through the existing class information. CIG is used to select the essential features.

Moreover, the CIG is computed using the following equation as,

\[
G(b, P) = X(p_b = 1, P) \times \log \frac{X(p_b = 1, P)}{X(p_b = 1)X(P)} + X(p_b = 0, Q) \times \log \frac{X(p_b = 0, Q)}{X(p_b = 0)X(Q)} \tag{1}
\]

\[
G(b, Q) = X(p_b = 1, Q) \times \log \frac{X(p_b = 1, Q)}{X(p_b = 1)X(Q)} + X(p_b = 0, P) \times \log \frac{X(p_b = 0, P)}{X(p_b = 0)X(P)} \tag{2}
\]

where, \(X(p_b = 1, G_b)\) represents the probability of \(b\) features, and \(X(p_b = 0, G_b)\) represents the probability of \(b\) feature which is absent from \(G_b\). \(P\) denotes the benign process and \(Q\) denotes the malicious process respectively.

CIG extracts the distinct OpCode with 4543 1-g and 610,109 2-g sequences. Here among the top most features,
nearly 82 features are selected and are represented as, \{b_{1},... ,b_{82}\}, where, \(b_{l}\) either belongs to 1-g or 2-g sequence. The selected features are limited to the size as, \(l = 82\), as there exists a considerable gap between \(b_{l} \leq 82\) and \(b_{l} \geq 83\) values. CIG ignores all the k-gram features, as its value is greater than 2. Based on the values of \(G(b,P)\) and \(G(b,Q)\) the features from the sequences are selected efficiently.

### 3.3 Attack detection using proposed MEHO-based stacked autoencoder classifier

The attack detection is done using the proposed MEHO-based stack autoencoder classifier, which is mandatory to restrict the unauthorised information access. Moreover, the attack detection is carried out using the attack detection techniques which aim to secure the data from intruders. Here the attack detection is performed using the proposed MEHO algorithm. The EHO algorithm effectively performs well in attack detection, but to enhance the security in the network resource, the parameters of the Moth algorithm are integrated with EHO to significantly generate better detection mechanism. The herding behaviour of the elephants is used to solve the optimisation problems. The algorithmic steps involved in the proposed algorithm are as follows:

(i) Population initialisation:

In the MEHO algorithm, various clans are collected together to form the elephant population, where each clan contains some specific number of elephants. At each generation, a group of male elephants exit from the elephant group and also live alone from their family. The elephants of the specific clans are grouped and live together under the matriarch leadership.

(ii) Clan updating operator:

The elephants live together under the matriarch leadership for each clan. Hence, the next position of each elephant present in the clan \(sk\) is predicted under the matriarch \(sk\). The elephant \(l\) present in the clan \(sk\) is updated using the equation below as,

\[
b_{\text{new,sk},l} = b_{sk,l} + \chi \times (b_{\text{best},sk} - b_{sk,l}) \times d
\]

where, \(b_{\text{new,sk},l}\) and \(b_{sk,l}\) are the newly updated and the old positions of the \(l\)th elephant in the \(sk\) clan respectively. \(\chi\) indicates the scale factor at the range of 0-1 which determines the matriarch \(sk\) on \(b_{sk,l}\). However, \(b_{\text{best},sk}\) indicates the matriarch \(sk\) which is the best fit elephant in the clan \(sk\). Moreover, the uniform distribution \(d\) is used when the value ranges from 0 to 1 respectively.

The attack detection is effectively performed by incorporating the moth flight feature of the MS algorithm into the EHO algorithm. Hence, the levy flight, phototaxis and the better searching ability of the moth flight boost up the functionality of the attack detection. Moreover, the herding behaviour of the elephant is integrated with the best moth fly to attain cyberattack detection.

The levy flight of the moth \(k\) is expressed as,

\[
b_{k}^{w+1} = \sigma \times (b_{k}^{w} + u \times (b_{\text{best}}^{w} - b_{k}^{w}))
\]

\[
\sigma = \frac{b_{k}^{w+1}}{b_{k}^{w} + u \times (b_{\text{best}}^{w} - b_{k}^{w})}
\]

Substitute the scale factor \(\sigma\) instead of \(\chi\) in Equation (3)

\[
b_{\text{new,sk},l} = b_{sk,l} + \frac{b_{k}^{w+1}}{b_{k}^{w} + u \times (b_{\text{best}}^{w} - b_{k}^{w})} \times (b_{\text{best,sk}} - b_{sk,l}) \times d
\]

Hence, the above equation can be rewritten as,

\[
b_{sk,l}(w + 1) = b_{sk,l}(w) + \frac{b_{sk,l}(w + 1)}{b_{sk,l}(w) + \nu(b_{\text{best}}(w) - b_{sk,l}(w))}
\]

\[
\times (b_{\text{best}}(w) - b_{sk,l}(w)) \times d
\]

\[
b_{sk,l}(w + 1) = b_{sk,l}(w) \left[\frac{b_{sk,l}(w) + \nu(b_{\text{best}}(w) - b_{sk,l}(w))}{b_{sk,l}(w) + \nu(b_{\text{best}}(w) - b_{sk,l}(w))} \right] \]

where, \(\nu\) represents the acceleration factor. The elephant with the fittest value in the clan is updated using the expression as,

\[
b_{\text{new,sk},l} = \lambda \times b_{\text{centr,sk}}
\]

where, \(\lambda\) is a scale factor with the value ranging from 0 to 1 which defines the influence of \(b_{\text{centr,sk}}\) on \(b_{\text{new,sk},l}\). In the above Equation (9), a new individual \(b_{\text{new,sk},l}\) is generated using the information of the elephants present in the clan \(sk\). \(b_{\text{centr,sk}}\) denotes the centre location of the clan \(sk\), and for the \(n\)th dimension, the updated position is represented as,

\[
b_{\text{centr,sk},n} = \frac{1}{j_{sk}} \sum_{l=1}^{j_{sk}} b_{sk,l,n}
\]

where, \(1 \leq n \leq D\) denotes the \(n\)th dimension, \(j_{sk}\) denotes the number of elephants present in the clan \(sk\), \(b_{sk,l,n}\) represents the \(n\)th individual elephant in \(b_{sk,l}\), and \(N\) denotes the total dimension. For \(sk\), the clan centre \(b_{\text{centr,sk}}\) is computed using the Equation (10).

(iii) Separating operator:

In the elephant family, when the male elephant reaches puberty, it will exit from the family and survive at some other
location. Such a process of separation is modelled using the separating operator, which is used to solve the optimisation problem. At each generation, the individual elephant with the worst fit value is used to compute the separating operator.

\[ b_{\text{sep}, sk} = b_{\text{min}} + (b_{\text{max}} - b_{\text{min}} + 1) \times m \]  

(11)

where, \( b_{\text{min}} \) and \( b_{\text{max}} \) denote the lower bound and the upper bound of the individual elephant. The worst elephant in the clan \( sk \) is indicated as \( b_{\text{sep}, sk} \). \( m \) denotes the uniform distribution and the stochastic distribution with the value ranging from 0 to 1. Algorithm (1) shows the pseudo-code of the proposed MEHO algorithm.

(a) Proposed MEHO-based stacked autoencoder classifier for attack detection

Autoencoder captures the selected features through the principle component analysis (PCA), and the autoencoder with the single layer does not have any directed loops. The encoder function uses the deterministic mapping to transform the input vector to the hidden vector. The input vector is encoded into the hidden representation using the autoencoder function, which is represented as,

\[ F = b(H_1K + D_1K) \]  

(12)

The hidden representation \( F \) is again decoded into \( K \) and is expressed as,

\[ K = a(H_3K + B_3K) \]  

(13)

**Algorithm 1 Pseudo code of the proposed MEHO algorithm**

| Input: Elephant population, \( S \) |
|------------------------------------|
| Output: Find the worst elephant \( b_{\text{sep}, sk} \) |
| Parameters: \( sk \rightarrow \) clan, \( j_{sk} \rightarrow \) Number of elephant present in the clan \( sk \), \( h_{l,b} \rightarrow \) Old position of the \( l \)th elephant in clan, \( h_{\text{new}, sk,l} \rightarrow \) New position of the \( l \)th elephant in clan, \( h_{\text{bst}, sk} \rightarrow \) matriarch \( sk \), \( mxgen \rightarrow \) maximum generation |

1 **Begin**
2 **While** \( w < mxgen \)
3 **Do**
4 Based on elephant fitness, the elephants are sorted
5 // **Clan updating operator** implementation
6 **for** \( sk = 1 \) **to** \( j_{\text{clan}} \)
7 **for** \( l = 1 \) **to** \( j_{sk} \)
8 The location \( h_{sk,l} \) is updated and \( h_{\text{new}, sk,l} \) is generated using \( h_{\text{new}, sk,l} = h_{sk,l} + \chi \times (h_{\text{bst}, sk} - h_{sk,l}) \times d \)
9 if \( h_{sk,l} = h_{\text{bst}, sk} \)
10 The location \( h_{sk,l} \) is updated and \( h_{\text{new}, sk,l} \) is generated using \( h_{\text{new}, sk,l} = \lambda \times h_{\text{gen}, sk} \)
11 **end if**
12 **end for** \( l \)
13 **end for** \( sk \)
14 // **Clan separating operator** implementation
15 **for** \( sk = 1 \) **to** \( j_{\text{clan}} \)
16 In the clan \( sk \), the worst elephant is replaced
17 **end for** \( sk \)
18 The population of the elephant is evaluated using the newly modified position
19 \( w = w + 1 \)
20 **end while**
21 **Terminate**

where, \( H_1 \) and \( H_2 \) denote the weight matrices, and \( D_1, D_2 \) indicate the bias vectors. The autoencoder function is trained using the backpropagation scheme for reducing the cost function and squared reconstruction error which is expressed using the equation below as,

\[ X(H,B) = \frac{1}{2E} \sum_{k=1}^{E} \left\| K_k^C - K_k^B \right\|^2 \]  

(14)

The overfitting is reduced by applying the scarcity constraints and the weights and the cost function is revealed using the equation below as,

\[ X(\theta) = \frac{1}{E} \sum_{k=1}^{E} \left\| K_k^C - K_k^B \right\|^2 + \sigma \left( \left\| H_1 \right\|^2 \right) \]  

(15)

\[ + \left\| H_2 \right\|^2 \right) + \delta \sum_{i=1}^{M} Y(K \mid K_i) \]  

(16)

where, \( p_{2,k} \) denotes the activation function of the hidden layer, and \( p_{2,k} \) represents the \( l \)th entry of the activation function.

The divergence of Kullback–Leibler is denoted as \( Y \) and is expressed as,

\[ Y(K \mid K_i) = K \log \frac{K}{K_i} + (1 - K) \log \frac{1 - K}{1 - K_i} \]  

(17)

where, \( Y(K \mid K_i) \) represents the sparsity condition, \( \left\| H_2 \right\|^2 \) and \( \left\| H_2 \right\|^2 \) are the parameter conditions, \( \sigma \) denotes the weight of the regularisation and \( \delta \) represents the weight of the sparse condition. \( K \) denotes the sparse parameter, \( K_i \) represents the average activation function of the hidden layer units and \( M \) indicates the number of units in the hidden layer.
The stacked autoencoder contains multiple layers, where each output layer is connected with the successive input layer. The output of the activation function using the stacked autoencoder is expressed as,

\[ p_{g,k}^e = b_g \left( p_{g-1,k}^e H_{g-1}^e, + \sum_k x_{g-1}^2 \right) \] (18)

Let us assume, \( p_{1,k} = c_k \), and \( d_{H,D}(c_k) = p_{H,k}^e \), hence, the cost function is represented as,

\[ X(H,D) = \frac{1}{2E} \sum_{k=1}^E \| d_{H,D}(c_k) - c_k \|^2 \] (19)

\[ X(\theta) = X(H,D) + \frac{\sigma}{2} \sum_{k=1}^{E} \sum_{k=1}^{b_k} \left( \sum_{c=1}^{H^b} H^b_k \right)^2 \]

\[ + \sum_{k=1}^{E} \sum_{c=1}^{K^S} \lambda^S Y \left( K^S \mid K^S \right) \] (20)

where, \( j_k \) indicates the total number of network layers, and \( \lambda^S \) and \( K^S \) denote the hyperparameters of the \( g \)th layer respectively. Figure 2 shows the classification model of the proposed MEHO-based stacked autoencoder. However, the stacked autoencoder consists of three different processes. At first, the parameters are initialised for each autoencoder, and in the next process, the activation function of the hidden layers is computed and in the third stage, the fine-tuning process is performed using the backpropagation model. Hence, the classification accuracy is enhanced by altering all the modelled parameters.

4 | RESULTS AND DISCUSSION

The results and discussions made using the proposed MEHO-based stacked autoencoder classifier are elaborated in this section. Moreover, the performance of the proposed approach is evaluated using the evaluation metrics, like DR, False Acceptance Rate (FAR) and accuracy, and the analysis is made using the existing methods respectively.

4.1 | Experimental setup

The proposed MEHO-based stacked autoencoder classifier is implemented in the MATLAB tool using the KDD cup dataset [33], and the NSL-KDD dataset [34]. The KDD cup dataset contains the standard set of information to be audited. The NSL-KDD dataset is the newest version of the KDD dataset, which is used to solve the inherent problem existing in the KDD dataset. The records present in the NSK-KDD datasets are reasonable test and train data. This feature makes the NSL-KDD dataset run the experimentation with the complete set without selecting a few portions randomly. In the training set, it does not contain any redundant records, hence the performance using the test set was enhanced. Table 1 represents the simulation setup of the proposed system.

4.2 | Evaluation metrics

The performance of the proposed approach is analysed and evaluated using the metrics, like accuracy, detection rate and FAR.

4.2.1 | Accuracy

It is defined as the measure of positive and negative predictions to the total number of measures.

\[ \Lambda = \frac{U + V}{\text{total}} \] (21)

where, \( U \) represents the total positive rate, and \( V \) indicates the total negative rate.

4.2.2 | Detection rate

It is defined as the ratio of true detection with the total positive and negative measure.

\[ K = \frac{U}{U + S} \] (22)

Here, \( S \) represents the false-negative rate.
with Hidden Neuron 6 is 0.8807 and MEHO with Hidden Neuron 8 is 0.8818, respectively. When training data = 70%, the accuracy obtained by the proposed MEHO with Hidden Neuron 2 is 0.8690, MEHO with Hidden Neuron 4 is 0.8772, MEHO with Hidden Neuron 6 is 0.8808 and MEHO with Hidden Neuron 8 is 0.8820, respectively. When training data = 80%, the accuracy obtained by the proposed MEHO with Hidden Neuron 2 is 0.8748, MEHO with Hidden Neuron 4 is 0.8782, MEHO with Hidden Neuron 6 is 0.8808 and MEHO with Hidden Neuron 8 is 0.8866, respectively. When training data = 90%, the accuracy obtained by the proposed MEHO with Hidden Neuron 2 is 0.8770, MEHO with Hidden Neuron 4 is 0.8804, MEHO with Hidden Neuron 6 is 0.8853 and MEHO with Hidden Neuron 8 is 0.8890, respectively. Therefore, it is clear that increasing the Hidden Neuron increases the accuracy of the proposed classifier.

Figure 3b shows the analysis of the detection rate with respect to the training data. When training data = 50%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.8433, MEHO with Hidden Neuron 4 is 0.8486, MEHO with Hidden Neuron 6 is 0.8799 and MEHO with Hidden Neuron 8 is 0.8884, respectively. When training data = 60%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.8464, MEHO with Hidden Neuron 4 is 0.8506, MEHO with Hidden Neuron 6 is 0.8819 and MEHO with Hidden Neuron 8 is 0.8918, respectively. When training data = 70%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.8465, MEHO with Hidden Neuron 4 is 0.8561, MEHO with Hidden Neuron 6 is 0.8832 and MEHO with Hidden Neuron 8 is 0.8954, respectively. When training data = 80%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.8471, MEHO with Hidden Neuron 4 is 0.8605, MEHO with Hidden Neuron 6 is 0.8882 and MEHO with Hidden Neuron 8 is 0.8956, respectively. It shows that increasing the Hidden Neuron significantly increases the detection rate. Figure 3c depicts the analysis of FAR with respect to the training percentage. When training data = 50%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.3400, MEHO with Hidden Neuron 4 is 0.3090, MEHO with Hidden Neuron 6 is 0.1693 and MEHO with Hidden Neuron 8 is 0.1642, respectively. When training data = 60%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.2396, MEHO with Hidden Neuron 4 is 0.2262, MEHO with Hidden Neuron 6 is 0.1683, and MEHO with Hidden Neuron 8 is 0.1365, respectively. When training data = 70%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.2355, MEHO with Hidden Neuron 4 is 0.2160, MEHO with Hidden Neuron 6 is 0.1680 and MEHO with Hidden Neuron 8 is 0.1361, respectively. When training data = 80%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.2216, MEHO with Hidden Neuron 4 is 0.2154, MEHO with Hidden Neuron 6 is 0.1680 and MEHO with Hidden Neuron 8 is 0.1361, respectively.
8 is 0.1303, respectively. When training data = 90%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.2147, MEHO with Hidden Neuron 4 is 0.2139, MEHO with Hidden Neuron 6 is 0.1548 and MEHO with Hidden Neuron 8 is 0.1439, respectively. It is clear that, when the number of Hidden Neuron increases, the FAR decreases significantly.

4.4.2 | Performance analysis using NSL-KDD dataset

Figure 4a shows the analysis of accuracy with respect to the training data. When training data = 50%, the accuracy obtained by the proposed MEHO with Hidden Neuron 2 is 0.6936, MEHO with Hidden Neuron 4 is 0.6999, MEHO with Hidden Neuron 6 is 0.8136 and MEHO with Hidden Neuron 8 is 0.8165, respectively. When training data = 60%, the accuracy obtained by the proposed MEHO with Hidden Neuron 2 is 0.8199, MEHO with Hidden Neuron 4 is 0.8211, MEHO with Hidden Neuron 6 is 0.8259 and MEHO with Hidden Neuron 8 is 0.8380, respectively. When training data = 70%, the accuracy obtained by the proposed MEHO with Hidden Neuron 2 is 0.8207, MEHO with Hidden Neuron 4 is 0.8304, MEHO with Hidden Neuron 6 is 0.8507 and MEHO with Hidden Neuron 8 is 0.8518, respectively. When training data = 80%, the accuracy obtained by the proposed MEHO with Hidden Neuron 2 is 0.8207, MEHO with Hidden Neuron 4 is 0.8304, MEHO with Hidden Neuron 6 is 0.8507 and MEHO with Hidden Neuron 8 is 0.8518, respectively. When training data = 90%, the accuracy obtained by the proposed MEHO with Hidden Neuron 2 is 0.8207, MEHO with Hidden Neuron 4 is 0.8304, MEHO with Hidden Neuron 6 is 0.8507 and MEHO with Hidden Neuron 8 is 0.8518, respectively. It is clearly depicted that increasing the number of Hidden Neurons increases the accuracy of the proposed classifier.

Figure 4b shows the analysis of the detection rate with respect to the training data. When training data = 50%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.6100, MEHO with Hidden Neuron 4 is 0.6295, MEHO with Hidden Neuron 6 is 0.8328 and MEHO with

![Figure 3](image-url) Performance analysis using KDD cup dataset, (a) accuracy, (b) detection rate and (c) FAR
Hidden Neuron 8 is 0.8451, respectively. When training data = 60%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.8194, MEHO with Hidden Neuron 4 is 0.8370, MEHO with Hidden Neuron 6 is 0.8656 and MEHO with Hidden Neuron 8 is 0.8797, respectively. When training data = 70%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.8798, MEHO with Hidden Neuron 4 is 0.8917, MEHO with Hidden Neuron 6 is 0.9016 and MEHO with Hidden Neuron 8 is 0.9052, respectively. When training data = 80%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.8840, MEHO with Hidden Neuron 4 is 0.8927, MEHO with Hidden Neuron 6 is 0.9038 and MEHO with Hidden Neuron 8 is 0.9121, respectively. When training data = 90%, the detection rate obtained by the proposed MEHO with Hidden Neuron 2 is 0.9072, MEHO with Hidden Neuron 4 is 0.9092, MEHO with Hidden Neuron 6 is 0.9233 and MEHO with Hidden Neuron 8 is 0.9251, respectively. Moreover, it is shown that increasing the number of Hidden Neurons significantly increases the detection rate.

Figure 4c depicts the analysis of FAR for the training percentage. When training data = 50%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.3470, MEHO with Hidden Neuron 4 is 0.3458, MEHO with Hidden Neuron 6 is 0.3343 and MEHO with Hidden Neuron 8 is 0.3341, respectively. When training data = 60%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.3432, MEHO with Hidden Neuron 4 is 0.3323, MEHO with Hidden Neuron 6 is 0.3074 and MEHO with Hidden Neuron 8 is 0.2935, respectively. When training data = 70%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.3418, MEHO with Hidden Neuron 4 is 0.3275, MEHO with Hidden Neuron 6 is 0.3032 and MEHO with Hidden Neuron 8 is 0.2844, respectively. When training data = 80%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.3184, MEHO with Hidden Neuron 4 is 0.3182, MEHO with Hidden Neuron 6 is 0.1550 and MEHO with Hidden Neuron 8 is 0.1550, respectively. When training data = 90%, the FAR obtained by the proposed MEHO with Hidden Neuron 2 is 0.1158, MEHO with Hidden Neuron 4 is 0.1095, MEHO with Hidden Neuron 6 is 0.1074 and MEHO with Hidden Neuron 8 is 0.0965, respectively. It is very clear that when the Hidden Neuron increases, the FAR rate decreases respectively.

4.5 Comparative analysis

This section elaborates on the comparative analysis of the proposed MEHO-based stacked autoencoder classifier using the metrics, like accuracy, detection rate and FAR respectively.
4.5.1 | Comparative analysis using KDD cup dataset

The comparative analysis made by the proposed classifier with respect to the evaluation metrics, such as accuracy, detection rate and FAR using the KDD cup dataset is shown in Figure 5. Figure 5a shows the analysis of accuracy with respect to the training data. When training data = 50%, the accuracy obtained by the proposed MEHO is 0.8575, while the percentage improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 1.83%, 6.70% and 704.37%, respectively. When training data = 60%, the accuracy obtained by the proposed MEHO is 0.8929, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 2%, 7% and 11% respectively. When training data = 70%, the accuracy obtained by the proposed MEHO is 0.8937, while the percentage improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 2.29%, 6.61% and 10.76% respectively. When training data = 80%, the accuracy obtained by the proposed MEHO is 0.895, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 2%, 7% and 12% respectively. When training data = 90%, the accuracy obtained by the proposed MEHO is 0.8978. When it is compared with the existing techniques, namely LSTM-RNN, DEL and DBN, the percentage of improvement is reported as 2%, 7% and 12% respectively.

Figure 5b shows the analysis of the detection rate by varying the training data. When training data = 50%, the accuracy obtained by the proposed MEHO is 0.8302, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 3.63%, 26.52% and 1.56% respectively. When training data = 60%, the detection rate obtained by the proposed MEHO is 0.9084, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 5%, 48% and 7% respectively. When training data = 70%, the accuracy obtained by the proposed MEHO is 0.9092, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 5.36%, 0.46% and 6.22% respectively. When training data = 80%, the accuracy obtained by the proposed MEHO is 0.9103, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 5.48%, 0.58% and 6.11% respectively. When training data = 90%, the detection rate obtained by the proposed MEHO is 0.9148. When it is compared with
the existing methods, namely LSTM-RNN, DEL and DBN, the percentage of improvement is reported to be 6%, 1% and 6%, respectively.

Figure 5c depicts the analysis of FAR with respect to the training data. When training data = 50%, the accuracy obtained by the proposed MEHO is 0.085, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 347.05%, 296.94% and 135.76%, respectively. When training data = 60%, the accuracy obtained by the proposed MEHO is 0.0850, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 347.05%, 187.05% and 120.47% respectively. When training data = 70%, the accuracy obtained by the proposed MEHO is 0.085, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 198.40%, 128.83% and 4.29% respectively. When training data = 90%, the accuracy obtained by the proposed MEHO is 0.0814, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 188.32%, 128.62% and 4.42% respectively.

4.5.2 | Comparative analysis using NSL-KDD dataset

The comparative analysis made by the proposed classifier with respect to the evaluation metrics, such as accuracy, detection rate and FAR using NSL-KDD dataset is shown in Figure 6. Figure 6a shows the analysis of accuracy with respect to the training data. When training data = 60%, the accuracy obtained by the proposed MEHO is 0.8426, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 6%, 5% and 0.33% respectively. When training data = 70%, the accuracy obtained by the proposed MEHO is 0.8439, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 6.22%, 5.17% and 0.48% respectively. When training data = 90%, the accuracy obtained by the proposed MEHO is 0.9286. When it is compared with the existing techniques, namely LSTM-RNN, DEL and DBN, the percentage of improvement is reported to be 2%, 2% and 7% respectively.

Figure 6b shows the analysis of the detection rate by varying the training data. When training data = 60%, the detection rate obtained by the proposed MEHO is 0.9170, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 47% and 1% respectively. When training data = 70%, the accuracy obtained by the proposed MEHO is 0.9172, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 5.47%, 0.23% and 0.52% respectively. When training data = 90%, the detection rate obtained by the proposed MEHO is 0.9258. When it is compared with the existing methods, namely LSTM-RNN, DEL and DBN, the percentage of improvement is reported as 6%, 1% and 1% respectively.

Figure 6c depicts the analysis of FAR with respect to the training data. When training data = 60%, the accuracy obtained by the proposed MEHO is 0.2948, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 188.32%, 128.62% and 10.04% respectively. When training data = 70%, the accuracy obtained by the proposed MEHO is 0.2894, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 9.46%, 20.80% and 3.80% respectively. When training data = 90%, the accuracy obtained by the proposed MEHO is 0.0636, while the percentage of improvement attained by the proposed MEHO classifier with respect to the existing methods, like LSTM-RNN, DEL and DBN is 44.65%, 108.17% and 283.96%, respectively.

4.6 | Comparative discussion

This section elaborates the comparative discussion made using the proposed MEHO-based stacked autoencoder classifier using the KDD and NSL-KDD dataset. Table 2 shows the comparative discussion of the proposed algorithm. In the KDD dataset, the proposed MEHO attained a better accuracy of 0.8978, which is 2.74%, 7.03% and 11.01% higher than the accuracy of the existing methods, like LSTM-RNN, DEL and DBN. In the NSL-KDD dataset, the detection rate obtained by the proposed MEHO is 0.9258, which is 6.32%, 1.08% and 1.08% better than the detection rate of the existing methods, like LSTM-RNN, DEL and DBN. The proposed lower FAR of 0.0636 was attained using the NSL-KDD dataset. From the results, it is exposed that the proposed method offers the best performance by obtaining the maximum accuracy, detection rate and minimum FAR than the existing methods. The reasons for the high performance of the proposed method are that it has the advantages of auto encoder, MS and EHO. The advantage of the stacked autoencoders is that they help in providing a similar image along with a reduced pixel value. Autoencoders also help to improve the performance of the data. EHO is used for solving continued optimisation problems and benchmark problems. MS can provide the best solutions more accurately. It is easy and flexible. So the proposed
method is better than the existing method. The time complexity of the proposed algorithm is $O(S \times D \times w)$, in which $S$ represents the population of elephants, $D$ represents the dimension and $w$ represents the position.

In the proposed method, the feature selection process is done by using the CIG which recognises the appropriate features based on available class information. Furthermore, the selection of accurate features helps to reduce the overfitting and to enhance the accuracy of the model. The EHO algorithm effectively performs well in attack detection, and the herding behaviour of the elephants is used to solve the optimisation problems. The Moth algorithm enhances the security in the network resources. Thus, the proposed MEHO-based stacked autoencoder approach effectively detects the cyber-attacks in the IoT devices.

## 5 CONCLUSION

The optimisation algorithm named MEHO-based stacked autoencoder is proposed in this research work to perform the attack detection in the IoT devices. The input data is preprocessed to remove the noise and artefacts, and hence the resultant data offers a high quality of information. The preprocessed data is allowed to the feature selection stage, where the class-aware features, like basic features, content-based

![Figure 6](image)

**FIGURE 6** Performance analysis using NSL-KDD dataset, (a) accuracy, (b) detection rate and (c) FAR

| Dataset   | Metrics/methods | LSTM-RNN | DEL   | DBN   | Proposed MEHO |
|-----------|------------------|----------|-------|-------|---------------|
| KDD cup   | Accuracy         | 0.8732   | 0.8346| 0.7989| 0.8978        |
|           | Detection rate   | 0.8604   | 0.905 | 0.8561| 0.9148        |
|           | FAR              | 0.2347   | 0.1861| 0.085 | 0.0814        |
| NSL-KDD   | Accuracy         | 0.9054   | 0.9065| 0.8645| 0.9286        |
|           | Detection rate   | 0.8672   | 0.9158| 0.9158| 0.9258        |
|           | FAR              | 0.092    | 0.1324| 0.2442| 0.0636        |

*TABLE 2* Comparative discussion
features and traffic-based features are selected using the CIG. The CIG overcomes the imperfect selection of global features. Finally, the attack detection is carried out based on the selected features using the stacked autoencoder classifier, which effectively classifies whether the traffic is an attack or not. The stacked autoencoder classifier is trained using the proposed MEHO algorithm, which is the combination of the MS and EHO algorithm. The implementation of the proposed approach is carried out using the NSL-KDD training dataset and the performance is evaluated using the metrics, namely, accuracy, FAR and detection rate, which acquired values of better accuracy as 0.9286, lower FAR as 0.0636 and better detection rate of 0.9258 respectively. In the future, the experimentation will be carried out using various new datasets with the new types of cyberattacks.

REFERENCES

1. Manti, S., et al.: Focus on pleiotropic role of HMGB1 in the onset of allergic and non-allergic respiratory diseases. Curr. Respir. Med. Rev. vol. 13, pp. 14–18 (2017)
2. George, A., Rajakumar, B. R.: On hybridising fuzzy min max neural network and firefly algorithm for automated coronary heart disease diagnosis. In: Fourth International Conference on Computing, Communications and Networking Technologies, Tiruchengode, India (2013)
3. Conti, L., et al. Simplified method for the characterization of rectangular straw bales (RSB) thermal conductivity. 245(5), 052035 (2017)
4. Vimolin, V, Vinusha, S: Edge-based image steganography using edge least significant bit (ELSB) technique, Multimed. Res. 1(1) (2018)
5. Kozik, R., et al.: A scalable distributed machine learning approach for attack detection in edge computing environments. J. Parallel Distrib. Comput. 119, 18–26 (2018)
6. Conti, L., Barbari, M., Monti, M.: Design of sustainable agricultural buildings: A case study of a wine cellar in Tuscany, Italy. Buildings 6(17) (2016)
7. Pacheco, J, Hariri, S: IoT security framework for smart cyber infrastructures. In: Proceedings of International Workshops on Foundations and Applications of Self Systems, pp. 242–247 (2016)
8. Pacheco, J, Tune, G, Hariri, S: Security framework for IoT cloud services. In: Proceedings of International Conference on Computer Systems and Applications, pp. 1–6 (2018)
9. Karatas, G., Demir, O, Sahingo, O.K.: Deep learning in intrusion detection systems. In: Proceedings of International Congress on Big Data: Deep Learning and Fighting Cyber Terrorism, pp. 113–116 (2018)
10. Elmaghraby, A.S, Losavio, M.M.: Cybersecurity challenges in Smart Cities: safety, security and privacy. J. Adv. Res. 5(4), 491–497 (2014)
11. Roopak, M., Tian, G.Y, Chambers, J: Deep learning models for cyber security in IoT networks. In: Proceedings of Annual Computing and Communication Workshop and Conference (2019)
12. Santi, A.S., et al.: Cybersecurity framework for internet of things-based energy internet. Future Generat. Comput. Syst. 93, 849–859 (2019)
13. Yan, J., et al.: Q-learning-based vulnerability analysis of smart grid against sequential topology attacks. IEEE Trans. Inf. Forensics Secur. 12(1), 200–210 (2017)
14. Badra, M, Zeadally, S: Design and performance analysis of a virtual ring architecture for smart grid privacy. IEEE Trans. Inf. Forensics Secur. 9(2), 321–329 (2014)
15. Saxena, N, Choi, B.J, Lu, R.: Authentication and authorisation scheme for various user roles and devices in smart grid. IEEE Trans. Inf. Forensics Secur. 11(5), 907–921 (2016)
16. Hu, J, Vasilakos, A.V.: Energy big data analytics and security: challenges and opportunities. IEEE Trans. Smart Grid. 7(5), 2423–2436 (2016)
17. Hu, B, Gharavi, H.: Smart grid mesh network security using dynamic key distribution with merkle tree 4-way handshaking. IEEE Trans. Smart Grid. 5(2), 550–558 (2014)
18. Chen, Q, Abdelwahed, S, Erradi, A.: A model-based validated autonomic approach to self-protect computing systems. IEEE Internet Things J. 1(5), 446–460 (2014)
19. Kim, K, Kim, I, Lim, J: National cybersecurity enhancement scheme for intelligent surveillance capability with public IoT environment. J. Super-comput. 73(3), 1140–1151 (2017)
20. Jiang, F, et al.: Deep learning based multi-channel intelligent attack detection for data security. IEEE Trans. Sustain. Comput. 5: 204–212 (2018)
21. Azmooodeh, A, Dehghantanha, A, Choo, K.K.R.: Robust malware detection for internet of battlefield things devices using deep eigenspace learning. IEEE Trans. Sustain. Comput. 4: 88-95 (2018)
22. Rathore, S, Park, J.H.: Semi-supervised learning based distributed attack detection framework for IoT. Appl. Soft Comput. 72, 79–89 (2018)
23. Diro, A.A, Chilamkurti, N.: Distributed attack detection scheme using deep learning approach for Internet of Things. Future Generat. Comput. Syst. 82, 761–768 (2018)
24. Aldaaj, A.: Enhancing cyber security in modern internet of things (IoT) using intrusion prevention algorithm for IoT (IPAI). IEEE Access, 1, 1 (2019)
25. Jasiul, B, Szpyrka, M, Sliva, J: Detection and modelling of cyber attacks with petri nets. Entropy, 16, 6603–6623 (2014)
26. Bojorje, P.D., et al.: A practical approach to detection of distributed denial-of-service attacks using a hybrid detection method. Comput. Electr. Eng. 73, 84–96 (2019)
27. Catak, FO, Mustazoglu, AF: Distributed denial of service attack detection using autoencoder and deep neural networks. J. Intell. Fuzzy Syst. 37, 1-11 (2019)
28. Wang, G.G.: Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Memetic Comput. 10, 1–14 (2018)
29. Wang, G.G., Deb, S, Coelho, L.D.S: Elephant herding optimization. In: Proceedings of 3rd International Symposium on Computational and Business Intelligence, pp. 1–5 (2015)
30. Jayapriya, K, Mary, N.A.B: Employing a novel 2-gramp subgroup intra pattern (2GSIP) with stacked auto encoder for membrane protein classification. Mol. Biol. Rep. 46(2), 2259-2272 (2019)
31. Wang, G-G, Deb, S, Coelho, LS: Elephant herding optimization. In: 3rd International Symposium on Computational and Business Intelligence (ISCBII), Bali, Indonesia (2015)
32. Wang, G-G, et al.: A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. Int. J. Bio-Inspired Comput. 8(6) (2016)
33. KDD Cup Dataset. https://www.kaggle.com/galaxyh/kdd-cup-1999-data. Accessed July 2019
34. NSL-KDD Dataset. https://www.unh.edu/cic/datasets/nsl.html. Accessed July 2019
35. Vojit, J: Deep neural networks and their implementation. Dissertation, Charles University in Prague (2016)

How to cite this article: Alotaibi AS. A hybrid attack detection strategy for cybersecurity using moth elephant herding optimisation-based stacked autoencoder. IET Circuits Devices Syst. 2021;15:224–236. https://doi.org/10.1049/cds.2.12016