International Journal of Basic and Applied Sciences, 14 (2) (2025) 577-590

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/dwfxsc41
Research paper

An Efficient Technique for Identifying Distributed Denial of Service Active Assaults Using Deep Neural Networks Based on the Adaptive System Intelligence Paradigm

Karthikeyan Kaliyaperumal [1]*, Raja Sarath Kumar Boddu [2], Sai Kiran Oruganti [3], Guidsa Tesema Kebesa [4], Mohsen Aghaeiboorkheili [5], Rajendran Bhojan [6]

[1] Post Doc Researcher, Lincoln University College, Malaysia
[2] Professor in CSE, Raghu Engineering College, Visakhapatnam, India.
[3] Professor, Faculty of Engineering and Built Science, Lincoln University College-KL Malaysia
[4] Dean, School of Informatics and Electrical Engineering, IoT – HH Campus, Ambo University, Ambo, Ethiopia
[5] Head of School, Mathematics and Computer Science, PNG University of Technology, Lae, Papua New Guinea
[6] School of Mathematics and Computer Science, PNG University of Technology, Lae, Papua New Guinea
**Corresponding author E-mail: pdf.kirithicraj@lincoln.edu.my*

Received: May 30, 2025, Accepted: June 24, 2025, Published: July 3, 2025

Abstract

A collection of interconnected devices that exchange data online is known as the Internet of Things. The IoT environment's diverse components make the distributed denial-of-service attack a security risk. One of the most important tasks in creating a smarter environment for end users is detecting DDoS attacks in the Internet of Things. A new version of the optimized Elman recurrent neural network (ERNN) is proposed to detect DDoS active attacks in Internet of Things scenarios. The proposed detection approach optimizes the weight and bias of ERNN (ABCO-ERNN) using a novel adaptive bacterial colony optimization (ABCO) technique. The ABCO algorithm uses an adaptable step size to increase the BCO's capacity for both exploration and exploitation. The four datasets, BoT-IoT, CIC-IDS2017, CIC-DDoS2019, and IoTID20, are used to compare performance, and five distinct performance measures, including accuracy, precision, sensitivity, specificity-ty, and f-measure, are considered. When compared to previous literature algorithms, the proposed ABCO-ERNN detection approach produced a high detection rate according to the experimental results.

Keywords: *Active Attacks; Detection; Deep Neural Network; DDoS; Optimization; Adaptive Learning.*

1. Introduction

The IoT is a network of physical "things" that have been equipped with software, sensors, and other tools enabling data sharing and communication across systems and devices. These technologies range from basic household goods to sophisticated industrial instruments [1], [2]. The Cisco IoT research team projected that 75.3 billion devices would be actively connected by 2025. One of the main goals of IoT devices is to make people's lives better, and they accomplish this with very little help from the user [3], [4]. It is challenging to uphold consistent communication standards across many networks because of the tremendous heterogeneity of the IoT and the difficulty of creating numerous solutions for every device. An intelligent system must therefore be capable of adapting to various devices while efficiently protecting them from security dangers. IoT devices also have various security weaknesses, which attackers may exploit to breach their security. The common IoT devices are vulnerable to attacks since their security measures are insufficient due to resource limits [5]. DDoS attacks on the IoT are viewed by researchers as a serious security issue. To slow down a network's performance and ultimately force it to fail, this attack floods it with fraudulent requests. If successful, the legitimate users of the network would experience a denial of service. Due to the significant cost of DDoS attacks, it is crucial to develop mitigation that can stop the attack and quickly identify the attacker(s) [6]. As a result, identifying DDoS attacks quickly and accurately is the key concern for network monitoring. DDoS attacks are growing more sophisticated as network technology continues to advance. Typically, it is necessary to continuously monitor and analyze network data to stop authorized users from breaking network policies and protect them from various attacks [7]. Machine learning (ML) algorithms can detect the key distinctions in abnormal data present within conventional data with high accuracy. Additionally, ML techniques have significant generalization potential, allowing them to recognize unidentified threats [8]. Deep learning (DL) is a type of ML algorithm that is a recent, successful technique in tackling a variety of real-world applications, including DDoS attack detection in IoT environments [4], [9]. Most traditional detection schemes now in use are ineffective and unsuitable for quickly detecting and mitigating IoT attacks. On the other hand, the ERNN is a well-known deep learning method and the fastest learning algorithm. However, a random selection of weights and bias may lead to the disturbing performance of ERNN and possibly fall into local optima. Thus, finding optimization techniques to train the network is required to increase its performance of the network. The population-based optimization algorithms have been applied to select the connection weights and bias of ERNN algorithms. However, the conventional procedures have fallen into many shortcomings, including convergence rate and local optima.

The Various bacterial foraging algorithms have gained increasing interest in recent years as a rich source of prospective engineering applications and computer models. A few models that replicate bacterial foraging behavior have been created and used to address various real-world issues. BCO is one of them, and a population-based approach for numerical optimization. BCO is a straightforward but effective optimization tool that replicates the foraging habits of E. coli bacteria. The original BCO algorithm, however, has a poor convergence tendency, and its search performance significantly decreases with the expansion of search space dimensionality and the problem complexity [10-15]. The chemotactic step size makes the balance between exploration and exploitation impossible to realize. Chemotaxis exhibits poor randomness due to bacteria's weak connections with one another. The abiotic community will arrive at the local convergence rather than the global one when searching in a complicated multimodal solution set due to the two limitations. To increase the convergence rate and prevent BCO's local optima problems, adaptive BCO is proposed. The adaptive step size is used to improve both the exploration and exploitation capability of the BCO, which is used for finding optimal connection weights and bias for ERNN.

The main goals of this study are to use a type of hybridization to increase the performance of ERNN classifiers to the global minima, speed up the learning process, and reduce error. The article then suggests a novel hybrid ABCO-ERNN model that uses the ABCO algorithm to decide on the appropriate classification percentage for the ERNN, optimize the weight, and minimize error. It has been discovered that the proposed ABCO-ERNN model is useful for global optimization while addressing various challenging classification issues. The aim of optimizing hyperparameters is crucial for achieving better generalization to unknown data, preventing overfitting or underfitting, increasing convergence speed, and improving model correctness. It can expedite convergence and enhance the ERNN's capacity for generalization. The following are the research work's contributions:

- The newly developed ABCO-ERNN detection method was created for detecting unusual incoming data
- The adaptive chemotaxis step size is used to improve the performance of BCO (ABCO)
- The ABCO is used to find the optimal weights and bias for the ERNN network
- Kolmogorov's theorem is used to pick the number of hidden neurons, and the learning rate is attained by trial and error to improve the efficiency of the DDoS attack detection method.
- Four datasets, including BoT-IoT, CIC-IDS2017, CIC-DDoS2019, and IoTID20, are used to assess the effectiveness of the suggested approach.
- For evaluating the performance of the ABCO-ERNN, five performance analyzers are taken into consideration. The relevant literature works summarize that ERNN is covered in the related works.

1.1. Deployment of ABCO-ERNN

Deploying ABCO-ERNN on resource-constrained IoT devices poses significant challenges due to limitations in energy, computation, and communication capabilities. To address these challenges, consider the following strategies:

1.2. Optimization techniques of ERNN

Model Pruning: Reduce the model's computational complexity by eliminating redundant neurons and connections, thereby decreasing energy consumption and improving inference speed. Knowledge Distillation: Train a smaller, more efficient model (student) to mimic the behavior of the larger ABCO-ERNN model (teacher), transferring knowledge while reducing computational requirements.

Quantization: Represent model weights and activations using fewer bits, reducing memory footprint and computational costs without significantly sacrificing accuracy.

1.3. Energy-efficient design

Energy Harvesting: Leverage energy harvesting techniques, such as solar or kinetic energy, to power IoT devices and reduce reliance on batteries. Low-Power Communication Protocols: Utilize low-power communication protocols like Bluetooth Low Energy (BLE) or LoRaWAN, which are designed for low data rates and long battery life. The Wake-Up Radio: Implement wake-up radio techniques to activate devices only when necessary. In federated learning and decentralized training is a train models are trained locally on edge devices and aggregate updates at the server, reducing communication costs and preserving data privacy, and selectively involve clients in the training process based on their available resources, ensuring efficient use of energy and computational capabilities.

By applying these strategies, you can optimize ABCO-ERNN for deployment on resource-constrained IoT devices, ensuring efficient and effective operation in real-world applications.

1.4. Overfitting mitigation

Overfitting mitigation is crucial in deep learning models, including those used for network intrusion detection. Here's a detailed explanation of advanced techniques beyond normalization and cross-validation. The dropout is a regularization technique that randomly sets a fraction of neurons to zero during training, preventing the model from relying too heavily on any single neuron or group of neurons. By randomly dropping neurons, the model is forced to learn multiple representations of the data, reducing overfitting and improving generalization. And Implementation. Dropout can be implemented in deep learning models using libraries like TensorFlow or PyTorch, typically with a dropout rate. Early Stopping is a technique that stops training when the model's performance on a validation set starts to degrade, preventing overfitting. By stopping training when the model starts to overfit, early stopping prevents the model from fitting the noise in the training data and improves generalization.

Regularization is a technique that adds a penalty term to the loss function to discourage large weights and prevent overfitting. The Regularization helps to reduce overfitting by adding a penalty term to the loss function, encouraging the model to find simpler solutions. In addition, techniques of data Augmentation involve generating new training data by applying transformations to existing data, helping to prevent overfitting by increasing the diversity of the training set. Batch normalization normalizes the inputs to each layer, helping to stabilize training and reduce overfitting. Methods involve combining multiple models to improve performance and reduce overfitting. By using these advanced techniques, you can improve the performance and robustness of your deep learning models for network intrusion detection, reducing the risk of overfitting and improving generalization to unseen data models and functions.

2. Related work

A major concern in the current computing industry is safeguarding IoT networks and infrastructure due to the alarmingly rapid rise in cyberattacks and security lapses. IoT devices run mostly without supervision and are less secure, which fully justifies the need for an attacker to create a botnet army to launch huge DDoS attacks[16]. Reviews of the literature are essential for comprehending the most recent methods, approaches, and discoveries concerning the identification of DDoS attacks in IoT environments. This section enables researchers to assess the performance of various algorithms and detection methods. They can evaluate the benefits and drawbacks of each strategy using the actual data that has been documented in the literature.
The current section highlights a few studies on DDoS attack detection in an IoT environment. B. Gupta et al. (2020) [17] provide an ML-based attack detection technique to recognize attack traffic in consumer IoT (CIoT). The suggested solution controls limited IoT network-specific features to authorize the local router to detect assaults with the aid of inexpensive ML classifiers. The suggested technique identifies packet-level attacks by looking at IoT network characteristics. A. K. Balyan et al. (2022) [18] developed an effective hybrid network-based intrusion detection system called HNIDS. The HNIDS combines the enhanced GA and PSO called EGA-PSO and improved random forest (IRF) to address the data-imbalance issue. The initial phase of the proposed HNIDS makes use of hybrid EGA-PSO algorithms to improve the minor data samples and create a balanced data set to more precisely learn the sample characteristics of small samples. A PSO is used in the recommended HNIDS to improve the vector. The addition of a multi-objective function to GA improves it by helping to investigate the key features, improve fitness outcomes, reduce dimensions, increase the true positive rate (TPR), and decrease the false positive rate (FPR). The IRF then removes the less important qualities, integrates a list of decision trees throughout each iterative process, monitors the performance of the classifier, and guards against overfitting problems in the subsequent phase.
The following review axioms are mentioned in the existing work of this research domain.

- Alsulami et al. (2022) [19] suggest a predictive ML approach to identify and categorize network movement. Our approach specifically distinguishes between regular and abnormal network activity. To evaluate how well five supervised learning models performed in classifying network activities for IoT devices. A. Mehmood et al. (2018) [20] developed an innovative method for applying the Naive Bayes classification algorithm to intrusion detection systems (IDSs). IDSs are established as multi-agent systems throughout the network to detect abnormal node activity and traffic. The fundamental ideas associated with our study and recent research in a related field are also included in the publication.
- Mihoub et al. (2022) [21] We were looking at DoS/DDoS attack detection for the IoT using ML techniques. Two parts are proposed: DoS/DDoS mitigation and detection. Because it identifies the accurate attack category and the packet category that was exploited in the attack, the detection module suggests fine-granularity recognition. A multi-class classifier using the "Looking-Back" idea is offered as the DoS/DDoS detection component, and it is tested using the Bot-IoT dataset.
- K. O. Adefemi Alimi et al. (2022)[22] proposed an IDS based on a redefined LSTM (RLSTM), detecting and classifying intrusions in IoT. To calculate the performance of the RLSTM, propose an RLSTM model that can identify DoS with a high degree of accuracy.
- N. F. Syed et al. (2020) [23] established a new MQTT protocol's application layer DoS attack detection framework, which is tested on real-world DoS attack scenarios that adhere to protocol regulations. Authors provide an MQTT protocol-specific ML-based detection mechanism to defend the MQTT message brokers from these kinds of threats. We test the efficacy of the suggested framework to identify these malicious attacks and show the effects of such assaults on different MQTT brokers through experiments. The collected results suggest that even in cases where MQTT brokers were denied legal access and resources were restricted, attackers are still able to overrun the server's resources. Furthermore, our identification of MQTT properties demonstrated good attack detection accuracy. The length-based and field-size-based characteristics are appropriate for identifying IoT-based attacks and significantly decrease the false-positive rates.
- O. Yousuf et al. (2022) [24] Have created a unique detection technique called DALCNN ("Detecting Attack using Live Capture Neural Network") to use the Open Day Light platform to establish a Software-defined-Network (SDN) and detect DDoS attacks in the IoT. To further categorize and detect DDoS attacks, a three-tier design is suggested. Using a unique activation function and DL principles, the algorithm categorizes the type of assault. To simulate a DDoS attack and exactly recognize the several DDoS attack types on the network, tools such as "Mininet and Wireshark" are used.
- Katib et al. (2023) [25] established a DL-based IDS called H3SC-DLIDS, which and a hybrid Harris Hawks optimization (HHO) with sine cosine algorithm (SCA) are designed for an IoT context backed by BC. The proposed H3SC-DLIDS technique seeks to detect the existence of DDoS attacks in the IoT with BC assistance. Blockchain technology (BC) is utilized to facilitate secure communication in IoT networks. The SCA is used for feature selection, and HHO for the creation of an H3SC approach are included in the proposed H3SC-DLIDS technique. An LSTM-Autoencoder (LSTM-AE) is used for the IDS process. Ultimately, the LSTM-AE's hyperparameter tweaking is accomplished by the implementation of the arithmetic optimization algorithm (AOA).
- S. Muthukumar et al. (2024) [26] developed an attention-aided DL methodology in a framework detection mechanism for DDoS attacks. The gathering of data from reputable web data sources is the main objective. Furthermore, utilizing a Deep Belief Network (DBN) and the suggested Enhanced Gannet Optimization Algorithm (EGOA), the important features from the collected data are extracted from the Deep Weighted Restricted Boltzmann Machine (RBM) with weights. In addition to improving network speed, this feature extraction process reduces dimensionality problems. Finally, for the aim of detecting DDoS threats, the obtained features are transmitted to the "Attention and Cascaded RNN with Residual LSTM called ACRNN-RLSTM" block model. This network's precise design enhances its trustworthiness by identifying sophisticated and novel threats. Y. K. Beshah et al. (2024) [27] present an adaptive system for detecting DDoS attacks online that uses various criteria often used in DDoS attack detection to identify and adjust to concept deviations in streaming data. To maximize zero-day DDoS detection and detect idea drift, this study also suggests a novel accuracy update weighted probability averaging ensemble (AUWPAE) method.
- P. Shukla et al. (2024) [28] developed a unique distributed ensemble technique for identifying deadly DDoS attacks. Using the amazing power of the H2O.ai distributed ML platform and the ensemble learning approach, this method consists of two main stages: the first is the development of a distributed ensemble method. Second, this technique was applied to the Apache Storm stream processing framework to quickly evaluate incoming network streams and classify them into eleven different groups, encompassing 10 different attack types and benign traffic, almost instantly.

- J. Bhayo et al. (2023) [29] introduce an ML-based method for identifying DDoS attacks in an SDN-WISE IoT controller. An ML-based detection module has been incorporated into the controller, and a testbed environment has been established to replicate the creation of DDoS attack traffic. The traffic is recorded via a logging method that was introduced to the SDN-WISE controller. Network.

```
Algorithm 1: BCO
Step 1: Set up the required parameters
Step 2: For each bacterium
Step 3: Chemotaxis and communication
Step 4: Reproduction and elimination
Step 5: Migration
Step 7: If the final stage is not reached, step 2 should be taken; if not, the process should be terminated.
Step 8: Maintain the most effective solutions in the final position
```

- Logs are written into a log file, which is subsequently pre-processed and transformed into a dataset. The SDN-WISE controller's ML DDoS detection module classifies SDN-IoT network packets using various ML algorithms. We compare the outcomes produced by the ML DDoS detection module and assess the effectiveness of the suggested framework using various traffic simulation situations.
- H. Lv et al. (2023)[30] suggest comparing the effectiveness of two methods: one that uses PCA and the other that does not. Preprocessing stages include the application of strong scaling and encoding algorithms. By combining PCA with Robust Scaler, the experiment results show a significant increase in the accuracy of DDoS attack detection in IoT devices. The goal of the current research was to address issues with scalability, efficiency, and efficacy in the mitigation of intrusions involving large-scale IoT datasets that were present in the literature described above.

2.1. Elman recurrent neural network (ERNN)

First created by Elman in 1990, the ERNN is a subset of the recurrent network model, including a configuration of different nodes. The architecture of ERNN is shown in Figure 1. Along with context units, the typical ERNN contains a levels arrayed horizontally. The input and hidden units' activations during the training phases are stored by BPNN operations in the ERNN's basic version. In ERNN, feedback is output to them by the hidden layer using a context layer. The feedback link can retain state information and display the completion times of input and output designs. Each layer of the ERNN contains one or more neurons that, to transfer information to the layer below, compute a nonlinear function of the total number of inputs with weights. The mathematical representation of the input layer corresponds to this.:

$$X_{it}(k) = \sum_{i=1}^{n} X_{it}(k-1) \tag{1}$$

Each neuron in the buried layer has the following input model:

$$\mathrm{net}_{jt}(k) = \sum_{i=1}^{n} W_{ij}X_{it}(k-1) + \sum_{j=1}^{p} u_j r_{jt}(k) \tag{2}$$

W_{ij}- weights value (input and hidden layers weight). C_j- weights (hidden and recurrent layers). The output of the hidden layer looks like this:

$$Z_{jt}(k) = f(\mathrm{net}_{jk}(k) \tag{3}$$

$$f(\mathrm{net}_{jk}(k) = \sum_{i=1}^{n} W_{ij}X_{it}(k-1) + \sum_{j=1}^{p} u_j R_{jt}(k) \tag{4}$$

The recurrent and output layers are reflected as follows:

$$R_{jt}(k) = Z_{jt}(k-1) \tag{5}$$

$$Y_t(k) = f(\sum_{j=1}^{p} V_j Z_{jt}(k) \tag{6}$$

2.2. Bacterial colony optimization (BCO)

Niu et al. (2012) presented the BCO evolutionary algorithm, a new type [31], Algorithm 1 shows its steps. BCO's biggest distinction from previous bacteria-inspired heuristic algorithms is that bacteria share information when searching for nutrition rather than swimming randomly. The phases of the BCO method are separated into three, including chemotaxis and communication, elimination and reproduction, and migration. Numerous practical applications have been solved with the help of the BCO technique. Communication is constant during the complete life cycle of a BCO in addition to chemotaxis. After extensive chemotaxis and communication, bacteria have two options. If they are unable to find food on their own, they may reproduce or starve to death. By pushing the envelope or searching for space, some people may put themselves in dangerous positions in a challenging environment. Throughout the elimination and reproduction phase, the high-energy bacteria will self-replicate to form new individuals, replacing the unhealthy ones. Bacteria with a lot of energy are good at finding nutrients. In the last stage, the bacteria might move inside the search range if specific requirements are satisfied. Chemotaxis and communication are used during the whole BCO process. Conversely, the other two phases are only performed under certain situations, such as after a predetermined number of cycles have been completed, the randomly generated number falls below a predetermined probability, etc. The length of the chemotaxis process can be described by two models, such as swimming and tumbling. Both an ideal and a chaotic seeking director have an impact on the direction of the search. The current positions of each bacterium are as follows:

$$\mathrm{Position}_i(T) = \mathrm{Position}_i(T-1) + C(i) * [f_i * (G_{\mathrm{best}} - \mathrm{Position}_i(T-1)) + (1-f_i) * (P_{\mathrm{best}_i} - \mathrm{Position}_i(T-1)) + \mathrm{turb}_i] \tag{7}$$

If there is no turbulence director present during the swimming process, the bacteria will move toward their ideal location, updating each other's positions as they go.

$$\mathrm{Position_i(T) = Position_i(T-1) + C(i) * [f_i * (G_{best} - Position_i(T-1)) + (1-f_i) * (P_{best_i} - Position_i(T-1))]} \quad (8)$$

Where, $\mathrm{turb_i}$ denotes turbulent direction variance value. $f_i \in \{0,1\}$. $\mathrm{P_{best}}$denotes the personal best. $\mathrm{G_{best}}$denotes the global best. $\mathrm{C(i)}$ is the value of the chemotaxis step size which is defined as follows,

$$C(i) = C_{\min} + \left(\frac{Iter_{\max} - Iter_j}{Iter_{\max}}\right)^n .(C_{\max} - C_{\min}) \quad (9)$$

Where, $\mathrm{Iter_{max}}$and $\mathrm{Iter_j}$ are the maximum and current iterations, respectively. n - symbolizes the linearly decreasing technique of the chemotaxis step.

2.3. Adaptive BCO (ABCO)

Any evolutionary and SI algorithms that search for promising areas in the problem's solution space will respond differently depending on how its parameters are adjusted. The way the parameters are configured directly affects how good the solution turns out, so it is important to understand what the most capable arrangement should be. At this point, fine-tuning such parameters becomes an optimization challenge within the problem being optimized [32], [33]. The BCO is a widely used SI algorithm for solving any optimization problem due to its global search ability. However, the original BCO algorithm has a poor convergence tendency, and its search performance significantly decreases with the expansion of search space dimensionality and the problem complexity. The main variable regulating how well the BCO approaches can search is the step size parameter $C(i)$. The following issues affect the BCO with a fixed step size:

1. If the step size is exceedingly small, it will require numerous chemotaxis steps to reach the optimal location. So, the rate of convergence also decreases [11], [34], [35]. For fewer iterations, it might not achieve the ideal position.
2. Although the bacteria rapidly reach the location of the optimal location, if the step size is very high, the precision becomes slow. For the remaining chemotaxis phases, it moves towards the maximum. In population-based optimization methods, effective global and local exploitation control is essential to identifying the best solution. To balance local and global searches, the adaptive step size is implemented in this research. As shown below, this function is expressed [36].

$$\mathrm{A = \frac{f^i(X)}{f^i_{max}}(x_{max} - x_{min})rand} \quad (10)$$

Where, $\mathrm{f^i(X)}$ is the $\mathrm{i^{th}}$ Bacterium fitness function value. $\mathrm{x_{max}}$ and $\mathrm{x_{min}}$are the maximum and minimum values of variables. The chemotaxis step length is adaptively adjusted during the running process of the algorithm to achieve a balance of local search and global optimization. The step length is adjusted adaptively as follows

$$\mathrm{C(i)_{new} = C(i)\ \frac{Iter_{max} - Iter_j}{Iter_{max}}.A} \quad (11)$$

Here, an adaptive updating method for the chemotaxis step size value is utilized to guarantee BCO convergence. Then, the final position of the best positions is considered as connection weights and biases of the ERNN. The following section discusses the proposed optimized ERNN method, which is applied to detecting DDOS attacks in IoT environments.

2.4. Proposed ABCO-ERNN

Finding the best learning rule, constructing the network's topology, and training the connection weights are the three primary focuses of ERNN optimization. The learning process of a model is controlled and monitored using the best parameters. These parameters are established before training can start, since they control the model's whole training process. They might not have an impact on the model's performance, but they undoubtedly have an impact on the quality and speed of the learning process. An effective model may be distinguished from an average model by carefully selecting its parameters. Parameter tuning, which is the process of choosing the ideal collection of parameters for a model, was carried out in this article. The weights and biases are initialized with ABCO in the first epoch, and those weights are then sent to the ERNN. The network's final cycle or epoch will be reached, or the MSE will be achieved, and ABCO will continue searching for the ideal weights. The best solution will be selected to update the weights in the subsequent cycle. In this study, the optimal value of the objective function is obtained by training the weights and biases of the ERNN using the ABCO approach. The objective function is defined as follows,

$$\mathrm{MSE} = \frac{1}{\mathrm{N}}(\mathrm{y_i} - \hat{y}_i)^2 \quad (12)$$

Where $\mathrm{y_i}$ is the forecasted value and $\hat{y}_i$s the actual value. The sample's length is N. Algorithm 2 shows the specific steps of the proposed method. Figure 2 shows the flowchart of the proposed ABCO-ERNN. In the ABCO-ERNN, set the hyperparameters of a population of bacterial agents at random at first. Analyze each hyperparameter set's first performance on the ERNN. Based on nutritional gradients (MSE), each bacterium modifies its position (hyperparameter set). Bacteria mimic local search by migrating toward areas with higher nutritional concentrations. Together, bacteria form groups and exchange information, facilitating collective movement toward the hyperparameter space's more promising areas. Exploration and exploitation are balanced in this era. Better-performing bacteria, or those with higher health, procreate and transfer their sets of hyperparameters. Rejecting fewer beneficial bacteria leads to a population that is more optimized. To keep the search space diverse and prevent local optima, randomly remove some bacteria and distribute new ones throughout it. This stage introduces diversity and aids in the exploration of new areas. Continue the chemotaxis, swarming, reproduction, and elimination-dispersal stages until the convergence requirements are satisfied, or for a predetermined number of rounds. Take the final population's bestperforming hyperparameter set and use it. One effective way to improve model performance in ERNN is to apply ABCO for hyperparameter adjustment. Through emulating the foraging behavior of bacteria, BCO offers a strong framework for investigating and taking advantage of the hyperparameter space, resulting in enhanced precision, quicker convergence, and superior generalization in sequential data challenges. Because ABCO is dynamic and adaptable, it is a good fit for optimizing complicated models, such as ERNNs, so that the network can reach its maximum potential.

3. Experimental results and analysis

The discussion and outcomes of experiments are crucial to the study of IoT DDoS attack detection because they offer practical support, understanding, and validation for suggested methods and approaches for detection. This section discusses the strength of the suggested ABCO-ERNN compared to several benchmark methods such as ABCO-ERNN, BCO-ERNN, BFO-ERNN, IPSO-ERNN, PSO-ERNN, GA-ERNN, ERNN, and BPNN. The compared algorithms were developed using MATLAB 2015R to acquire numerical values. In this study, four different datasets—BoT-IoT, CIC-IDS2017, CIC-DDoS2019, and IoTID20—as well as five different performance indicators, are used to evaluate the performance of comparison algorithms. 20 trials are conducted on each dataset to validate these methods.

a) Problem statement

DDoS attack detection is a significant difficulty in a cloud environment and is categorized into two groups: benign and attacks. The suggested attack detection system employs the ABCO-ERNN classifier. Each sample in the groups is classified as either normal or under attack. The suggested detector is based on a supervised classifier, and it must be trained before it can be used to identify attacks. An extensive dataset of labeled samples is used to train the classifier. The output O is calculated after the group of samples X is applied to the trained classifier in groups. The model's output classifies each sample as either an attack or a typical sample. The sample belongs to the normal class if $\mathrm{t} = [0,1]$ and the attack class if $\mathrm{t} = [1,0]$.

b) Datasets

- BoT-IoT: UNSW built the BoT-IoT by establishing an undisturbed network environment. Normal traffic and traffic from botnets were kept apart. To facilitate the tagging procedure, the data files were divided into an attack group and a subgroup based on protocols [37]. Our suggested framework was tested using the Bot-IoT dataset, which contains both typical IoT network traffic and a range of attacks. “DDoS, DoS, OS and Service Scan, Keylogging, and Data Exfiltration Attacks” are all included in the dataset with more than 72,0,00,000 data samples and 43 features.
- CIC-IDS2017 [38]To assess our work, data on both regular and DDoS assaults were collected across five days in 2017, throughout a variety of cyberattacks and non-attack days [38]. It comprises label attributes, 80 network flow features, and 225,742 samples containing both attack and normal data. Due to the uneven nature of this dataset, we altered the training dataset for this study such that the proportion of attack and normal data was balanced, resulting in a reduction of the number of instances to 80 characteristics. There are two types of datasets: harmful networks and regular networks. (“SSH, FTP, HTTP, HTTP, and email protocols”).
- CIC-DDoS2019: It is entirely considered normal and malicious (“DNS, LDAP, MSSQL, NetBIOS, NTP, SNMP, SSDP, SYN, TFTP, and UDP”) [39, 40]. To anticipate DDoS attacks, it has 50,063,112 instances with 80 features and 11 class labels. CICFlowMeter is a widely used tool for creating network traffic characteristics.
- IoTID20: It imitates a contemporary IoT network communication trend and is one of the few publicly available IoT intrusion detection datasets, which are its two main advantages. The most recent iteration of the IoTID20 dataset comprises 86 network features, three label features, and 6,25,783 samples [41]. There are two different categories: abnormality and normal. Four distinct attacks fall under the category of anomalies: DoS, “SYN flooding, brute force, HTTP and UDP flooding, MITM, and scan (Host port and port OS)”.
- Dataset Limitations
  - Class Imbalance: Many datasets suffer from class imbalance, where one class (e.g., normal traffic) has a significantly larger number of instances than the other class (e.g., attack traffic). This can lead to biased models that perform well on the majority class but poorly on the minority class.
  - Evolving Attack Patterns: Network attacks are constantly evolving, and datasets may not reflect the latest attack patterns. This can lead to models that are not effective against new or unknown attacks.
  - Lack of Diversity: Datasets may not capture the diversity of real-world network traffic, including different protocols, applications, and user behaviors.
  - Labeling Errors: Datasets may contain labeling errors, where normal traffic is mislabeled as attack traffic or vice versa. This can lead to biased models and inaccurate performance evaluations.
  - Potential Biases in datasets may be biased towards specific types of networks, protocols, or applications, which can limit their generalizability to other contexts. In confirmation bias, researchers may unintentionally introduce bias into the dataset or model development process by selectively focusing on certain types of attacks or features.

c) Data splitting

The entire initial data set is randomly divided into k equal-sized subsamples using the k-fold cross-validation technique [42]. Due to its simplicity in implementation and comprehension, as well as the fact that the output results typically have a smaller bias in skill estimates, it is frequently utilized in classification tasks. The following succinctly describes the standard k-fold cross-validation procedure: The randomly shuffled data set is divided into k groups; use the group as a test data set or hold out for each distinct k grouping. You can use the remaining groupings as a training set of data. Using this strategy, every sample can be used once in the test set and then used $\mathrm{k} - 1$time to train the model [43]. Using tenfold cross-validation (9:1), the experiments were designed to train and test using nine of the k subset data sets for training and the remaining dataset for testing.

d) Preprocessing

The present section discusses the preprocessing of the datasets, such as removing the inappropriate data samples, such as NaN and INF. Furthermore, samples with the NaN and INF feature values were removed from the dataset. Since the datasets from CIC-IDS2017 and CIC-DDoS2019 contain various socket information, including Source/Destination IP, Source/Destination Port, and flow ID. We removed socket-involved features from the data samples to address the overfitting problem because they can vary throughout networks. The final dataset comprises 77 & 78 different features, except for the traffic label for the CIC-IDS2017 & CIC-DDoS2019, respectively. During training, the anomaly detection model may be biased toward more frequent records due to the duplication of records in the dataset. We only kept one copy of each entry and removed all duplicate records from the data to fix the issue. After the procedure, the sample count of the CIC-IDS2017 (DoS) dataset is reduced to 587,966. The process of data normalization converts the original vector of data into a new vector whose normalization equals one. The main advantages of data normalization are improved object-to-data mapping, fewer inconsistent data instances, and enhanced consistency. Data must normalize the values of the features within the range between 0 and 1 when using the min-max approach [44], [45], which is defined as follows,

$$z = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{13}$$

Where z - result of the normalization, x - normalized value, max(x) - upper value and min(x) smallest value of the dataset.

e) Parameter optimization for ERNN

Hyperparameters of an ERNN form the model's learning process and performance, such as the number of hidden neurons, weights, and bias, learning rate, activation function, and Epochs. Random hyperparameter selection is an unsatisfactory option for tuning ERNNs due to its inefficiency, low coverage, lack of convergence, noisy performance, scaling concerns, and lack of insight, even though it is simple to apply and occasionally produces adequate results. Hence, the selection of hyperparameters of ERNN is a significant and more difficult task. The significance of the hyperparameters is discussed as follows,

f) Parameter setting for BCO

Using the right algorithm parameters can greatly improve a solution's performance. The swim step (N_s) and chemotaxis step (N_C)*They* are used to calculate the convergence rate of BCO. To ascertain the ideal value of the chemotaxis step size, the current article concentrated on the adaptive method. The elimination and dispersal probability (P_{ed}) values are important BCO parameters. The best probability value determines the local optimum problem. Table 1 lists the parameters of the BCO and ERNN algorithms.

Table 1: Parameter's Values

BCO		ERNN	
Parameter(s)	Values	Parameters	Values
Maximum Iterations	500	Objective function	MSE
Population size	50	Learning rate	0.6
N_C	100	Epochs	1000
N_s	4	Target error	0.0005
N_{re}	4	Weight range	-1 and 1
P_{ed}	0.25		

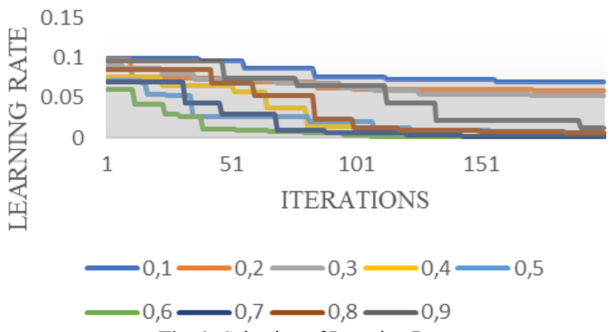

Fig. 1: Selection of Learning Rate.

g) Performance analyzer

To assess system performance, the following metrics were used: accuracy, sensitivity, specificity, precision, and F-Score.

- Accuracy: the proportion of all cases to the number of correctly classified abnormal and normal cases which is defined as follows,

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \tag{14}$$

- Sensitivity (Recall): The ratio of the total number of anomalous instances to the number of successfully classified abnormal instances which is defined as follows,

$$\text{Sensitivity} = \frac{TP}{TP+FN} \tag{15}$$

- Specificity, which is the proportion of cases that were genuinely negative but were nevertheless categorized as such, can be computed as follows:

$$\text{Specificity} = \frac{TN}{FP+TN} \tag{16}$$

- Precision: the ratio of successfully diagnosed attack samples to all instances classified as attacks which is defined as follows,

$$\text{Precision} = \frac{TP}{TP+FP} \tag{17}$$

- F-Score: The model's performance is expressed as the harmonic average of its precision and recall measures which is defined as follows,

$$F - \text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{18}$$

A successful prediction of attack traffic is referred to as a true positive (TP). The ability to anticipate ordinary traffic properly is referred to as a true negative (TN). Inaccurate predictions of regular traffic are known as false positives (FP). Results of attack flows that are erroneously predicted to be negative (FN).

h) Compared with existing methods

The performance of the ABCO-ERNN compared with recent optimized approaches such as BCO-ERNN, BFO-ERNN, IPSO-ERNN, PSO-ERNN, GA-ERNN, ERNN, and BPNN. A training network technique that has gained more traction is the BPNN. Nevertheless, there are two major problems with the BPNN method: sluggish convergence and instability. Compared to BPNN, ERNN is a faster learning method. However, BPNN learns ERNN. They arise from the anticipation of surpassing the minimum of the error surface and the potential to become stuck in a local minimum. ERNN plays a direct role in the convergence of the process of BP learning. So, generally speaking, the major problem with the BPNN technique and the ERNN was that it may become stranded in local minima as it learned and propagated over the network [47]. To deliver derivative-free solutions for difficult classification problems, several SI techniques are combined with ERNN training procedures. The GA [48], PSO [49], IPSO [50], BFO [51], and BCO [52] approaches have been proposed to optimize the ERNN methods. The connection weights or structure of the ERNN are optimized using the GA. Such connection weight and threshold optimization can increase learning effectiveness. The GA technique, however, is prone to local optima and has very poor solution accuracy [53].

The PSO is a well-known SI approach based on fish schooling behavior to solve the drawbacks of local optima [47]. However, it falls into premature convergence and poor generalization ability. The self-learning and collective learning of particles are governed by learning variables. An asynchronous regulation technique for learning factors is proposed to significantly enhance the convergence rate and reduce the local optimization. However, both PSO and IPSO cannot speed up the expected convergence and again fall into premature convergence. Recent years have seen a substantial increase in the use of bacteria foraging-based techniques like BFO and BCO to address premature convergence and local optima issues. However, the convergence rate of BFO and BCO is very low [11]. Similarly, chemotaxis is the most crucial computational method used by the BCO while searching the optimization space since it has an important impact on how the algorithm is explored and used [54]. The random chemotaxis step size is disturbing the performance of BCO, such as a low convergence rate and low solution accuracy. Hence, the suggested model uses adaptive chemotaxis step value to enhance the convergence rate of BCO called ABCO, which is used learning process of ERNN for obtaining optimal weights and bias.

i) Results analysis

The training and testing phases are utilized to generate performance comparison findings for all detection methods. Testing and training are critical steps in developing DDoS attack detection systems for IoT environments. It comprises various critical steps required to ensure the detection algorithms' dependability, efficacy, and correctness. This section discusses the performance of the proposed ABCO-ERNN approach for DDoS detection in an IoT environment. The research work analyses the various factors to prove the ability of the ABCO-ERNN detection method, such as performance in the training and testing phase, convergence analysis, and computation time for training and testing. Further, the gathered datasets are used to train the detection algorithms during the training phase. This entails providing labeled data to the algorithms and classifying DDoS attack traffic as malicious and regular traffic as benign. Based on the labels supplied, the algorithms learn to differentiate between benign and malevolent traffic patterns. They also modify the parameters of these models to enhance detection accuracy and optimize efficiency. The detection algorithms are tested on various datasets that were not used during training after they have been trained. The accuracy of the trained models in detecting DDoS attacks is evaluated during this testing phase, with an emphasis on reducing false positives and false negatives. Therefore, the current study focused on training and testing outcomes to assess the effectiveness of the established DDoS assault technique.

Detailed experimental research is conducted on four distinct datasets to determine the effectiveness of the suggested methods. The number of biases and weights for each unique component of an ERNN structure is comparable to the problem dimensions. To find the optimal classification percentage, reduce error, and optimize the weights and biases of the ERNN, the ABCO is proposed. The optimal sample weighting of the vectors and biases is one of the several model parameters that the SI technique is used to find. The present section displays the performance results of comparable detection techniques for all datasets. Two kinds of performance analysis are conducted, namely training and testing.

The datasets are separated using a 10-fold cross-validation method. The training performance is shown in Tables 2,3,4, and 5, the graphical representation is shown in Figures 4,5, 6, and 7, and the testing performance is shown in Figures 8,9,10, and 11. Table 2 and Figure 4 display training performance comparisons for the BoT-IoT dataset. The proposed ABCO-ERNN method obtained the values of an accuracy of 96.61 %, Sensitivity of 97.20 %, Specificity of 96.20 %, Precision of 96.17 %, and F-score of 96.59 %. The other algorithms, namely BCO-ERNN, BFO-ERNN, IPSO-ERNN, PSO-ERNN, GA-ERNN, ERNN, and BPNN, have accuracy values that are significantly lower than those of the suggested algorithms. For example, these other algorithms' accuracy values are 95.63 %, 93.38 %, 87.45 %, 82.22 %, 80.12 %, 77.17 %, and 75.84 %; their sensitivity values are 96.70 %, 95.03 %, 88.45%, 83.89%,81.48%, 78.74%, and 76.78%; their specificity values 94.60%,91.84%,86.49%,80.71%, 78.87%, and 75.77%,; their precision values are 94.4891.55%, 86.14%, 79.76%,77.96%,74.45%, and 74.09%; their F-Score values 95.58%,93.25%, 87.28%,81.77%, 79.68%,76.53%,75.41% which is significantly less than the suggested ABCO-ERNN algorithms. The complete results show that the suggested ABCO-ERNN technique outperforms the other detection methods.

Table 2: Training Performance Results for the BoT-IoT

Approaches	Accuracy	Sensitivity	Specificity	Precision	F-Score
ABCO-ERNN	96.61	97.02	96.20	96.17	96.59
BCO-ERNN	95.63	96.70	94.60	94.48	95.58
BFO -ERNN	93.38	95.03	91.84	91.55	93.25
IPSO -ERNN	87.45	88.45	86.49	86.14	87.28
PSO -ERNN	82.22	83.89	80.71	79.76	81.77
GA-ERNN	80.12	81.48	78.87	77.96	79.68
ERNN	77.17	78.74	75.77	74.45	76.53
BPNN	75.84	76.78	74.97	74.09	75.41

Table 3: Training Performance Results for the CIC-IDS2017

Approaches	Accuracy	Sensitivity	Specificity	Precision	F-Score
ABCO-ERNN	98.85	99.93	97.81	97.77	98.84
BCO-ERNN	97.04	99.26	95.01	94.79	96.97
BFO -ERNN	96.20	98.22	94.35	94.12	96.12
IPSO -ERNN	95.03	96.81	93.39	93.14	94.94
PSO -ERNN	94.07	95.50	92.72	92.49	93.97
GA-ERNN	93.14	94.47	91.89	91.65	93.04
ERNN	92.06	93.29	90.90	90.64	91.95
BPNN	91.23	92.28	90.23	89.99	91.12

Table 4: Training Performance Based on CICDDOs 2019

Approaches	Accuracy	Sensitivity	Specificity	Precision	F-Measure
ABCO-ERNN	98.89	99.99	97.83	97.79	98.88
BCO-ERNN	96.49	97.92	95.14	94.99	96.43
BFO -ERNN	93.43	94.27	92.62	92.48	93.36
IPSO -ERNN	92.89	93.72	92.09	91.94	92.82
PSO -ERNN	90.35	90.99	89.72	89.56	90.27
GA-ERNN	87.74	88.35	87.14	86.94	87.64
ERNN	84.24	85.89	82.73	81.94	83.87
BPNN	82.03	82.86	81.25	80.78	81.80

Table 5: Training Performance Results for IoTID20

Approaches	Accuracy	Sensitivity	Specificity	Precision	F-Measure
ABCO-ERNN	98.74	99.49	98.01	97.98	98.73
BCO-ERNN	96.95	97.45	96.46	96.42	96.93
BFO-ERNN	93.38	94.82	92.04	91.78	93.27
IPSO-ERNN	91.14	92.10	90.21	89.99	91.03
PSO -ERNN	89.04	89.51	88.57	88.43	88.97
GA-ERNN	87.23	88.03	86.47	86.19	87.10
ERNN	85.82	86.49	85.16	84.89	85.68
BPNN	83.05	83.43	82.68	82.49	82.96

Table 3 and Figure 5 display training performance comparisons for the CIC-IDS2017 dataset. The proposed ABCO-ERNN method obtained the value of accuracy of 99.85%, Sensitivity of 99.93%, Specificity of 97.81%, Precision of 97.77%, and F-Score of 98.84%. The other algorithms, namely BCO-ERNN, BFO-ERNN, IPSO-ERNN, PSO-ERNN, GA-ERNN, ERNN, and BPNN, have accuracy values that are significantly lower than those of the suggested algorithms. The complete results show that the suggested ABCO-ERNN technique outperforms the other detection methods. Table 4 and Figure 3 display performance comparisons for the CIC-DDoS2019 dataset. The proposed ABCO-ERNN method obtained the value of accuracy of 98.89%, Sensitivity of 99.99%, Specificity of 97.83%, Precision of 97.79%, and F-Score of 98.88%. Similarly, Table 5 and Figure 4 display performance comparisons of various algorithms for the IoTID20 dataset. The proposed ABCO-ERNN method obtained the value of accuracy of 98.74%, Sensitivity of 99.49%, Specificity of 98.01%, Precision of 97.78%, and F-Score of 98.73. Based on the training phase, the performance of the suggested ABCO-ERNN provided good accuracy when compared with some standard detection methods and some variants of ERNN algorithms, according to the study of the overall results.

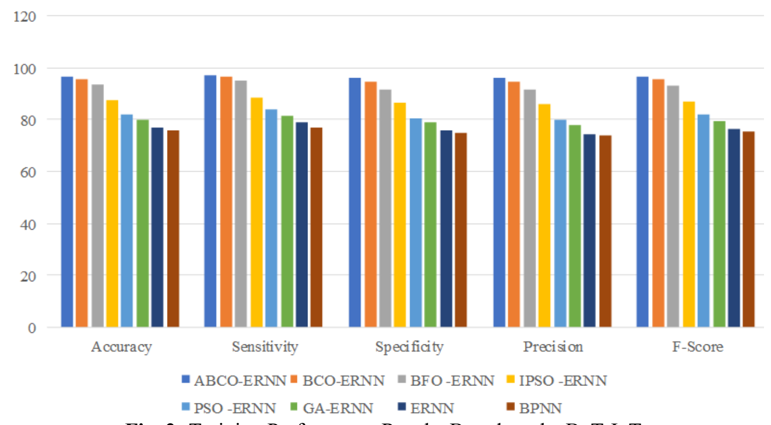

Fig. 2: Training Performance Results Based on the BoT-IoT.

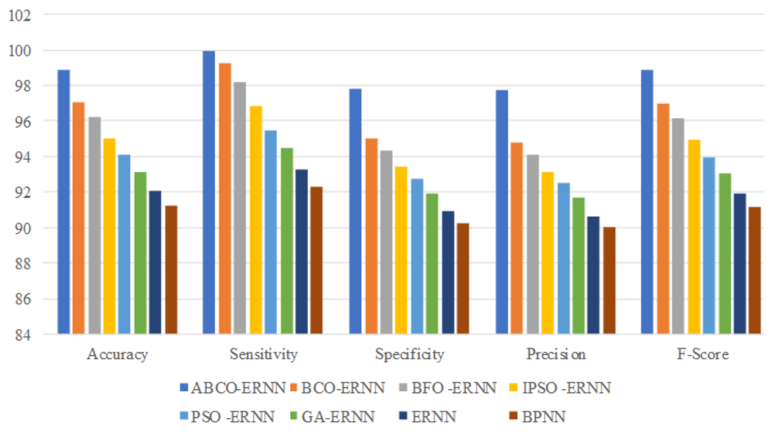

Fig. 3: Training Performance Results Based on the CIC-IDS2019.

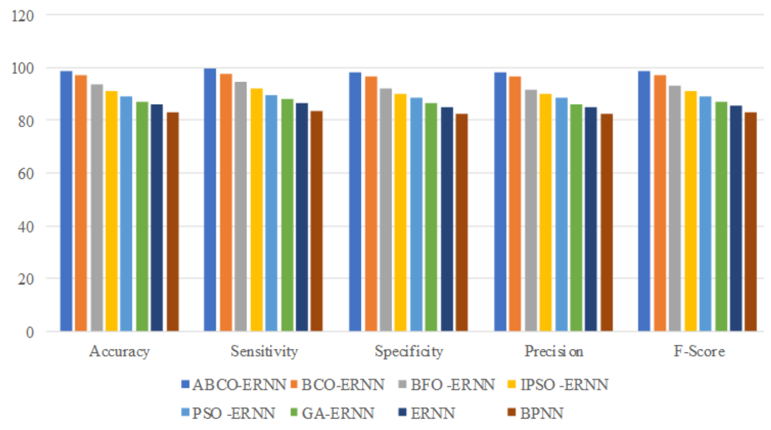

Fig. 4: Training Performance Based on the IoTID20 Dataset.

Table 6: Testing Performance Results for the BoT-IoT

Approaches	Accuracy	Sensitivity	Specificity	Precision	F-Score
ABCO-ERNN	97.63	98.95	96.39	96.29	97.60
BCO-ERNN	95.16	95.51	94.82	94.79	95.14
BFO -ERNN	93.31	94.35	92.32	92.14	93.23
IPSO -ERNN	88.30	89.21	87.43	87.14	88.16
PSO -ERNN	83.14	84.24	82.10	81.53	82.86
GA-ERNN	80.71	81.93	79.57	78.79	80.33
ERNN	78.19	79	77.42	76.79	77.88
BPNN	76.33	77.02	75.67	75.06	76.02

Table 7: Testing Performance Results for the CIC-IDS2017

Approaches	Accuracy	Sensitivity	Specificity	Precision	F-Score
ABCO-ERNN	99.21	99.98	98.46	98.44	99.20
BCO-ERNN	98.21	99.98	96.55	96.43	98.17
BFO -ERNN	97.38	98.74	96.10	95.99	97.34
IPSO -ERNN	96.07	97.58	94.65	94.48	96.01
PSO -ERNN	95.2	96.54	93.93	93.76	95.12
GA-ERNN	94.15	94.88	93.45	93.34	94.10
ERNN	93.20	93.87	92.55	92.44	93.15
BPNN	92.84	93.80	91.92	91.75	92.76

Table 8: Testing Performance Based on CICDDoS 2019

Approaches	Accuracy	Sensitivity	Specificity	Precision	F-Measure
ABCO-ERNN	99.46	99.62	99.29	99.29	99.45
BCO-ERNN	97.71	98.96	96.52	96.43	97.68
BFO -ERNN	95.89	96.85	94.98	94.88	95.85
IPSO -ERNN	92.46	93.76	91.24	90.98	92.35
PSO -ERNN	91.47	92.73	90.28	89.99	91.34
GA-ERNN	90.83	92.06	89.67	89.37	90.69
ERNN	87.20	88.03	86.41	86.12	87.06
BPNN	84.12	85.29	83.02	82.46	83.85

Table 9: Testing Performance Results for IoTID20

Approaches	Accuracy	Sensitivity	Specificity	Precision	F-Measure
ABCO-ERNN	99.14	99.49	98.79	98.78	99.13
BCO-ERNN	97.03	97.76	96.32	96.27	97.00
BFO-ERNN	95.21	96.20	94.26	94.14	95.16
IPSO-ERNN	93.39	94.36	92.46	92.29	93.31
PSO -ERNN	90.49	92.00	89.08	88.69	90.31
GA-ERNN	89.08	89.64	88.54	88.38	89.00
ERNN	87.26	88.04	86.52	86.24	87.13
BPNN	85.01	85.39	84.64	84.48	84.93

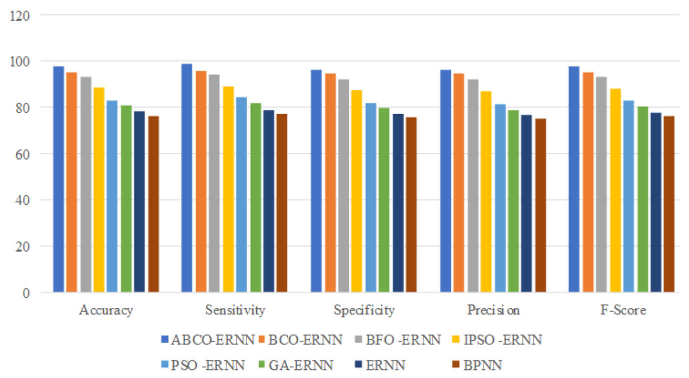

Fig. 5: Testing Performance Based on BoT-IoT.

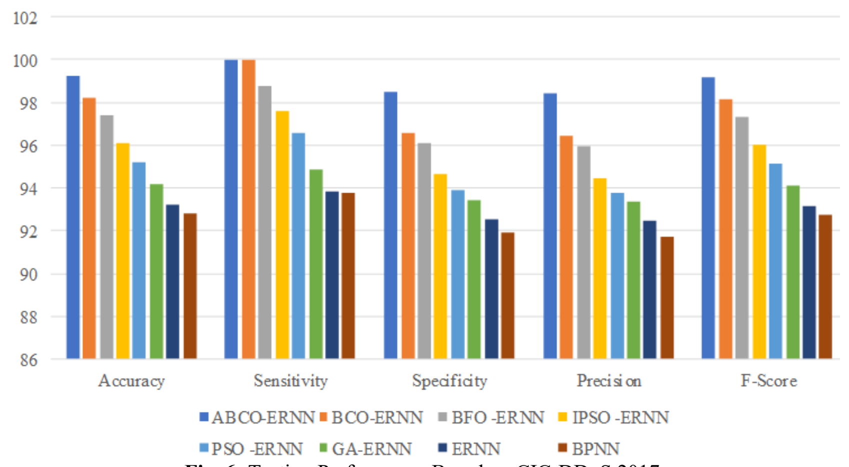

Fig. 6: Testing Performance Based on CIC-DDoS 2017.

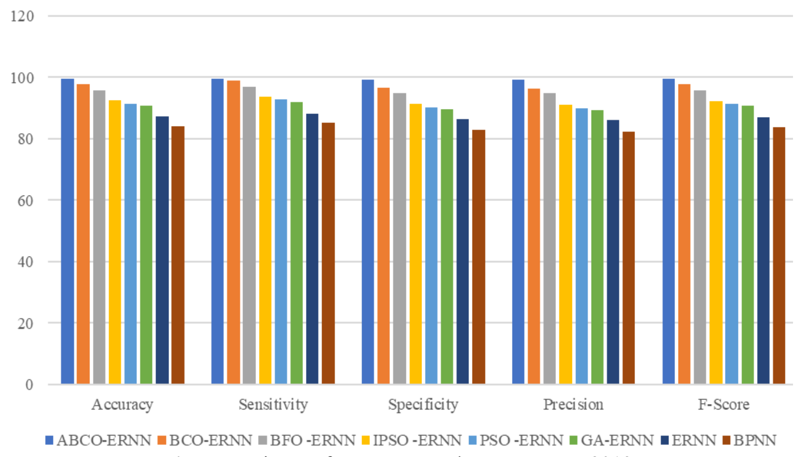

Fig. 7: Testing Performance Based on CIC-DDoS2019.

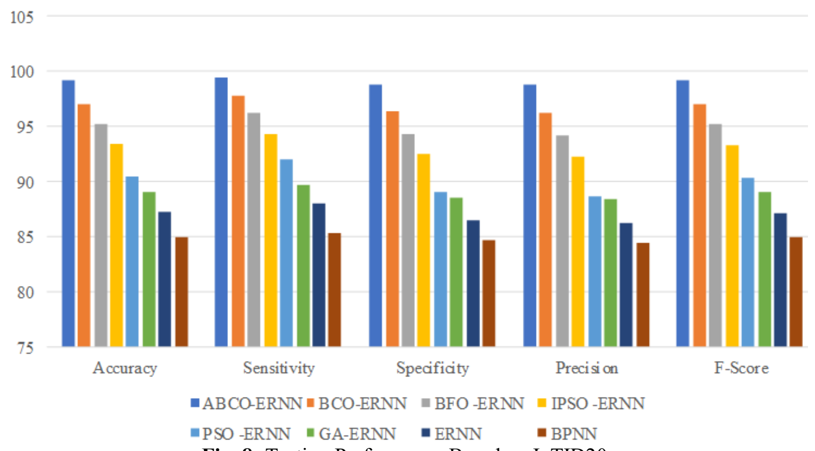

Fig. 8: Testing Performance Based on IoTID20.

The testing phase revealed that the developed ABCO-ERNN outperformed current detection techniques in terms of detection accuracy according to Figures 5,6, 7 and 8. The developed method produced high detection accuracy for all datasets. For example, the value of detection accuracy of BoT-IoT is 97.63%, CIC-IDS2017 is 99.21%, CIC-DDoS2019 is 99.46%, and IoTID20 is 99.14% are obtained. Similarly, other performance metrics such as sensitivity, specificity, precision, and f-measure also show the best performance in all datasets. Tables 10 and 11 show the computation time for the training and testing phases, and the graphical representation is shown in Figures 9 and 10. Elman RNNs rely heavily on computational time to detect DDoS attacks. Achieving a balance between resource usage and model performance requires effective time management. However, finding optimal hyperparameters of ERNN can manage the computational time. Hence, the proposed ABCO-ERNN uses the ABCO for finding proper weights and bias, the trial-and-error method is used for finding the appropriate learning rate, and Kolmogorov's method is used to find the optimal number of hidden neurons. So, the combination of these three methods makes an optimal ERNN network that can produce a minimal time to detect DDoS. The study comparing training and testing times shows that the suggested ABCO-ERNN generated a quicker training time across all datasets when compared with other detection methods. Convergence analyses of the suggested ABCO-ERNN are compared with some variants of ERNN, such as BCO-ERNN, BFO-ERNN, and IPSO-ERNN, as shown in the Figures. The proposed ABCO-ERNN approach produced a very low MSE with BoT-IoT, CIC-IDS2017, and CIC-DDoS2019 datasets. However, the ABCO-ERNN approach produced the slightly lowest MSE with the IoTID20 datasets. IoT DDoS attack detection systems are developed and evaluated in large part using convergence analysis. Convergence analysis sheds light on the detection algorithm's stability and rate of convergence. It contributes to the computation of the algorithm's detection and reaction times to DDoS attacks while reducing false positives and false negatives.

Table 10: Training Time Comparisons (Sec.)

Methods	BoT-IoT	CIC-IDS2017	CIC-DDoS2019	IoTID20
ABCO-ERNN	169.58	131.32	152.09	172.81
BCO-ERNN	174.67	137.94	165.28	179.24
BFO-ERNN	181.59	143.66	170.92	195.46
IPSO-ERNN	197.37	159.18	183.04	205.54
PSO -ERNN	215.87	164.95	197.68	216.81
GA-ERNN	239.67	179.67	205.85	231.67

Table 11: Testing Time Comparisons (Sec.)

Methods	BoT-IoT	CIC-IDS2017	CIC-DDoS2019	IoTID20
ABCO-ERNN	15.85	8.94	14.41	12.38
BCO-ERNN	20.67	13.34	22.68	18.97
BFO-ERNN	32.94	19.54	30.18	24.18
IPSO-ERNN	40.72	29.48	38.52	35.39
PSO -ERNN	48.28	38.75	45.67	44.27
GA-ERNN	56.46	46.71	52.18	54.94

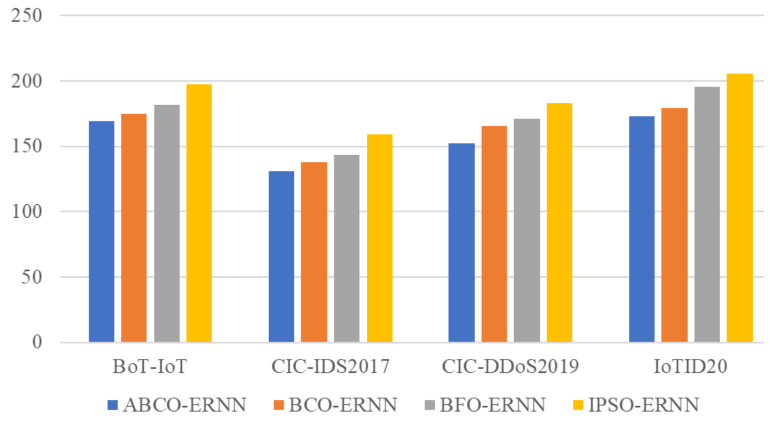

Fig. 9: Training Computation Time Comparison.

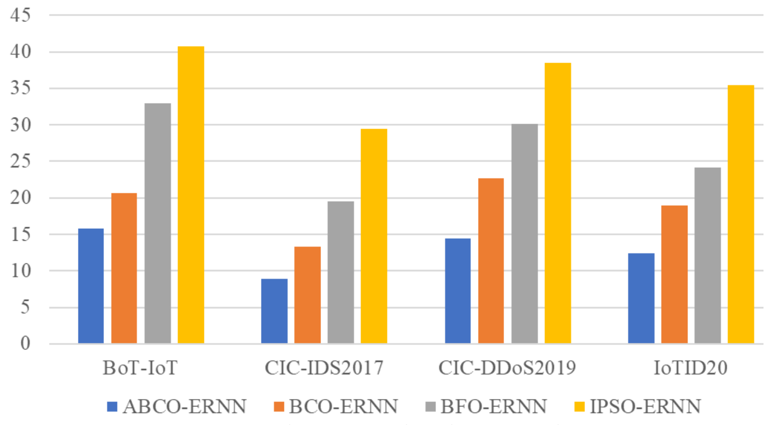

Fig. 10: Testing Computation Time Comparison.

Since ABCO facilitates the process of identifying the best configurations to increase the precision, rate of convergence, and generalization of ERNN models by effectively navigating and utilizing the hyperparameter space. ERNN hyperparameter optimization with ABCO offers a reliable way to improve model performance. By utilizing the collective intelligence of bacteria to traverse the intricate world of hyperparameters, this method produces improved DDoS attack detection results and greater model tweaking. Overall, the training and testing results demonstrate that the ABCO-ERNN model performs better than the previous model. To increase detection accuracy and speed of convergence, the ABCO's global capacity is used to optimize the ABCO-ERNN weights and bias. In a similar vein, the number of hidden neurons and learning rate are also tweaked to improve ERNN performance more effectively. Kolmogorov's theorem is used to select several hidden neurons in the hidden layer, and the trial-and-error method is used to determine the learning rate. The suggested ABCO-ERNN convergence with minimal MSE when compared with other approaches on four active attack datasets, according to the convergence study.

4. Future enhancements and directions

To enhance the future work section, consider these concrete research questions and practical enhancements:

- Adapting to Zero-Day Attacks: How can machine learning models be designed to detect and respond to previously unknown zero-day attacks, minimizing damage and downtime?
- Integration with Emerging Technologies: What are the potential benefits and challenges of integrating network intrusion detection systems with emerging technologies like blockchain, edge computing, or the Internet of Things (IoT)?
- Explainability and Transparency: How can explainable AI (XAI) techniques be applied to network intrusion detection systems to improve transparency and trust in AI-driven decision-making?

- Real-Time Threat Intelligence: Developing a real-time threat intelligence system that can share actionable insights across organizations and industries to enhance collective security.
- Hybrid Intrusion Detection: Designing a hybrid intrusion detection system that combines the strengths of signature-based, anomaly-based, and machine learning-based approaches to improve detection accuracy and reduce false positives.
- Cloud-Based Scalability: Investigating cloud-based scalability solutions for network intrusion detection systems to handle increasing network traffic and data volumes.
- Potential Application Domains
- Industrial Control Systems: Applying network intrusion detection techniques to industrial control systems (ICS) to protect critical infrastructure from cyber threats.
- Smart Cities: Developing smart city applications that leverage network intrusion detection systems to enhance public safety and security.
- Cybersecurity Training: Creating cybersecurity training programs that incorporate network intrusion detection systems to educate professionals about threat detection and response[1].
- Data Collection: Efforts should be made to collect and label large, diverse datasets that reflect real-world network traffic and attack patterns.
- Data Sharing: Datasets should be shared and made publicly available to facilitate research and development of more effective network intrusion detection models.
- Model Evaluation: Models should be evaluated using a variety of metrics, including accuracy, precision, recall, and F1-score, to provide a comprehensive understanding of their performance.
- By acknowledging and addressing these limitations and biases, researchers and practitioners can develop more effective and robust network intrusion detection models that can detect and respond to evolving attack patterns in real-world networks.

5. Conclusion

In this study, the improved ERNN model is suggested for detecting DDoS active attack contexts. The weights and bias vectors of the ERNN neurons are optimized by the ABCO for enhanced detection rate and fast convergence. To make a more efficient DDoS attack detection method, the learning rate is achieved using a trial-and-error method, and several hidden neurons in the hidden layer are selected based on Kolmogorov's theorem. The training and testing that were run were designed to assess how well the ABCO-ERNN performed, considering four well-known DDoS attack datasets. During the training and testing phases, the accuracy, consistency, and resilience of each model are assessed. Overall, the test results show that while executing several DDoS attack datasets such as BoT-IoT, CIC-IDS2017, CIC-DDoS2019, and IoTID20, the ABCO-ERNN model outperforms the other approaches. The experimental results showed that, in terms of detection rate and convergence speed, the recommended detection ABCO-ERNN performed better than similar detection systems. Future research should consider investigating any other nature-inspired algorithms. The model is then trained, and the idea of quantum computing is introduced to see what impact it has on the training procedure.

6. Declarations

Ethical Approval: Not applicable
Competing interests: The authors have disclosed no conflicts of interest.
Funding: No organizations or financial sources are helping to fund this study.
Availability of data and materials: Online datasets can be downloaded from the NSL-KDD Web Links

References

[1] S. V. K, G. K. V, A. K. D, P. L. B, and Y. B, "AOA based Masked Region-CNN model for Detection of Parking Space in IoT Environment," *International Research Journal of Multidisciplinary Technovation,* vol. 6, no. 1, pp. 97-108, 01/27 2024, https://doi.org/10.54392/irjmt2418.
[2] F. Ullah *et al.*, "Cyber security threats detection in internet of things using deep learning approach," *IEEE access,* vol. 7, pp. 124379-124389, 2019. https://doi.org/10.1109/ACCESS.2019.2937347.
[3] R. Kavitha, K. Saravanan, S. A. Jebakumari, and K. Velusamy, "Machine learning algorithms for IoT applications," in *Artificial Intelligence for Internet of Things*: CRC Press, 2022, pp. 185-214. https://doi.org/10.1201/9781003335801-11.
[4] D. N, J. Katiravan, and S. S.P, "Botnet Attack Detection in IoT Devices using Ensemble Classifiers with Reduced Feature Space," *International Research Journal of Multidisciplinary Technovation,* vol. 6, no. 3, pp. 274-295, 05/22 2024, https://doi.org/10.54392/irjmt24321.
[5] S. K. V, M. K. V, C. N. Azmea, and K. K. Vaigandla, "BCSDNCC: A Secure Blockchain SDN framework for IoT and Cloud Computing," *International Research Journal of Multidisciplinary Technovation,* vol. 6, no. 3, pp. 26-44, 04/16 2024, https://doi.org/10.54392/irjmt2433.
[6] N. K. Muthunambu, S. Prabakaran, B. PrabhuKavin, K. S. Siruvangur, K. Chinnadurai, and J. Ali, "A Novel Eccentric Intrusion Detection Model Based on Recurrent Neural Networks with Leveraging LSTM," *Computers, Materials & Continua,* vol. 78, no. 3, 2024. https://doi.org/10.32604/cmc.2023.043172.
[7] E. Gyamfi and A. Jurcut, "Intrusion Detection in Internet of Things Systems: A Review on Design Approaches Leveraging Multi-Access Edge Computing, Machine Learning, and Datasets," *Sensors,* vol. 22, no. 10, p. 3744, 2022. https://doi.org/10.3390/s22103744.
[8] R. Raphael and P. Mathiyalagan, "Intelligent Hyperparameter Tuned Deep Learning based Android Malware Detection and Classification Model," *Journal of Circuits, Systems and Computers,* 2022. https://doi.org/10.1142/S0218126623501918.
[9] K. Vijayakumari and V. Baby Deepa, "Fuzzy C-Means Hybrid with Fuzzy Bacterial Colony Optimization," in *Advances in Electrical and Computer Technologies: Select Proceedings of ICAECT 2020*, 2021: Springer Singapore, pp. 75-87. https://doi.org/10.1007/978-981-15-9019-1_7.
[10] K. Tamilarisi, M. Gogulkumar, and K. Velusamy, "Data clustering using bacterial colony optimization with particle swarm optimization," in *2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 2021: IEEE, pp. 1-5. https://doi.org/10.1109/ICECCT52121.2021.9616695.
[11] H. Wang, L. Tan, and B. Niu, "Feature selection for classification of microarray gene expression cancers using Bacterial Colony Optimization with multi-dimensional population," *Swarm and Evolutionary Computation,* vol. 48, pp. 172-181, 2019. https://doi.org/10.1016/j.swevo.2019.04.004.
[12] B. Niu, T. Xie, Y. Bi, and J. Liu, "Bacterial colony optimization for integrated yard truck scheduling and storage allocation problem," in *International Conference on Intelligent Computing*, 2014: Springer, pp. 431-437. https://doi.org/10.1007/978-3-319-09330-7_50.
[13] B. Niu, Q. Liu, Z. Wang, L. Tan, and L. Li, "Multi-objective bacterial colony optimization algorithm for integrated container terminal scheduling problem," *Natural Computing,* vol. 20, no. 1, pp. 89-104, 2021. https://doi.org/10.1007/s11047-019-09781-3.

[14] B. Bala and S. Behal, "AI techniques for IoT-based DDoS attack detection: Taxonomies, comprehensive review and research challenges," *Computer science review,* vol. 52, p. 100631, 2024. https://doi.org/10.1016/j.cosrev.2024.100631.

[15] Kaliyaperumal, D. K., Boddu, P. R. S. K. ., & Oruganti, P. S. K. . (2025). An analysis of alternative machine learning and deep learningalgorithms for categorization and detection of various active network assaults. *International Journal of Basic and Applied Sciences*, *14*(1), 414-421. https://doi.org/10.14419/zywhgb37

[16] S. S. Babu and K. Jayasudha, "A simplex method-based bacterial colony optimization for data clustering," in *Innovative Data Communication Technologies and Application: Proceedings of ICIDCA 2021*: Springer, 2022, pp. 987-995https://doi.org/10.1007/978-981-16-7167-8_72.

[17] B. Gupta, P. Chaudhary, X. Chang, and N. Nedjah, "Smart defense against distributed Denial of service attack in IoT networks using supervised learning classifiers," *Computers & Electrical Engineering,* vol. 98, p. 107726, 2022. https://doi.org/10.1016/j.compeleceng.2022.107726.

[18] A. K. Balyan *et al.*, "A hybrid intrusion detection model using ega-pso and improved random forest method," *Sensors,* vol. 22, no. 16, p. 5986, 2022. https://doi.org/10.3390/s22165986.

[19] A. A. Alsulami, Q. Abu Al-Haija, A. Tayeb, and A. Alqahtani, "An Intrusion Detection and Classification System for IoT Traffic with Improved Data Engineering," *Applied Sciences,* vol. 12, no. 23, p. 12336, 2022. https://doi.org/10.3390/app122312336.

[20] B. Sivasakthi and D. Selvanayagi, "Prediction of Osteoporosis Disease Using Enhanced Elman Recurrent Neural Network with Bacterial Colony Optimization," in *Computational Vision and Bio-Inspired Computing: Proceedings of ICCVBIC 2022*: Springer, 2023, pp. 211-220https://doi.org/10.1007/978-981-19-9819-5_16.

[21] A. Mihoub, O. B. Fredj, O. Cheikhrouhou, M. Krichen, "Denial of service attack detection and mitigation for internet of things using looking-back-enabled machine learning techniques," *Computers & Electrical Engineering,* vol. 98, p. 107716, 2022. https://doi.org/10.1016/j.compeleceng.2022.107716.

[22] K. O. Adefemi Alimi, K. Ouahada, A. M. Abu-Mahfouz, and O. A. Alimi, "Refined LSTM Based Intrusion Detection for Denial-of-Service Attack in Internet of Things," *Journal of Sensor and Actuator Networks,* vol. 11, no. 3, p. 32, 2022. https://doi.org/10.3390/jsan11030032.

[23] N. F. Syed, Z. Baig, A. Ibrahim, and C. Valli, "Denial of service attack detection through machine learning for the IoT," *Journal of Information and Telecommunication,* vol. 4, no. 4, pp. 482-503, 2020. https://doi.org/10.1080/24751839.2020.1767484.

[24] O. Yousuf and R. N. Mir, "DDoS attack detection in Internet of Things using recurrent neural network," *Computers and Electrical Engineering,* vol. 101, p. 108034, 2022. https://doi.org/10.1016/j.compeleceng.2022.108034.

[25] I. Katib and M. Ragab, "Blockchain-assisted hybrid harris hawks optimization based deep DDoS attack detection in the IoT environment," *Mathematics,* vol. 11, no. 8, p. 1887, 2023. https://doi.org/10.3390/math11081887

[26] S. Muthukumar and A. Ashfauk Ahamed, "A novel framework of DDoS attack detection in network using hybrid heuristic deep learning approaches with attention mechanism," *Journal of High Speed Networks,* no. Preprint, pp. 1-27, 2024. https://doi.org/10.3233/JHS-230142.

[27] Y. K. Beshah, S. L. Abebe, and H. M. Melaku, "Drift Adaptive Online DDoS Attack Detection Framework for IoT System," *Electronics,* vol. 13, no. 6, p. 1004, 2024. https://doi.org/10.3390/electronics13061004

[28] J. Bhayo, S. A. Shah, & D. Draheim, "Towards a machine learning-based framework for DDOS attack detection in software-defined IoT (SD-IoT) networks," *Engineering Applications of Artificial Intelligence,* vol. 123, p. 106432, 2023. https://doi.org/10.1016/j.engappai.2023.106432

[29] H. Lv, Y. Du, X. Zhou, W. Ni, and X. Ma, "A Data Enhancement Algorithm for DDoS Attacks Using IoT," *Sensors,* vol. 23, no. 17, p. 7496, 2023. https://doi.org/10.3390/s23177496.

[30] B. Niu and H. Wang, "Bacterial colony optimization," *Discrete Dynamics in Nature and Society,* vol. 2012, 2012. https://doi.org/10.1155/2012/698057.

[31] R. Li and Z.-J. Hu, "A self-adaptive particle swarm optimisation and bacterial foraging hybrid algorithm," *International Journal of Wireless and Mobile Computing,* vol. 11, no. 3, pp. 258-265, 2016. https://doi.org/10.1504/IJWMC.2016.081157.

[32] S. S. Babu and K. Jayasudha, "A simplex method-based bacterial colony optimization algorithm for data clustering analysis," *International Journal of Pattern Recognition and Artificial Intelligence,* vol. 36, no. 12, p. 2259027, 2022. https://doi.org/10.1142/S0218001422590273.

[33] V. Prakash, V. Vinothina, K. Kalaiselvi, and K. Velusamy, "An improved bacterial colony optimization using opposition-based learning for data clustering," *Cluster Computing,* vol. 25, no. 6, pp. 4009-4025, 2022. https://doi.org/10.1007/s10586-022-03633-z.

[34] K. Velusamy and R. Amalraj, "Performance of the cascade correlation neural network for predicting the stock price," in *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 2017: IEEE, pp. 1-6. https://doi.org/10.1109/ICECCT.2017.8117824.

[35] V. Kůrková, "Kolmogorov's theorem and multilayer neural networks," *Neural networks,* vol. 5, no. 3, pp. 501-506, 1992. https://doi.org/10.1016/0893-6080(92)90012-8.

[36] M. F. Ab Aziz, S. A. Mostafa, C. F. M. Foozy, M. A. Mohammed, M. Elhoseny, and A. Z. Abualkishik, "Integrating Elman recurrent neural network with particle swarm optimization algorithms for an improved hybrid training of multidisciplinary datasets," *Expert Systems with Applications,* vol. 183, p. 115441, 2021. https://doi.org/10.1016/j.eswa.2021.115441.

[37] A. Sadeghi-Niaraki, P. Mirshafiei, M. Shakeri, and S.-M. Choi, "Short-term traffic flow prediction using the modified elman recurrent neural network optimized through a genetic algorithm," *IEEE Access,* vol. 8, pp. 217526-217540, 2020. https://doi.org/10.1109/ACCESS.2020.3039410.

[38] K. Xie, H. Yi, G. Hu, L. Li, and Z. Fan, "Short-term power load forecasting based on Elman neural network with particle swarm optimization," *Neurocomputing,* vol. 416, pp. 136-142, 2020. https://doi.org/10.1016/j.neucom.2019.02.063.

[39] L. Yang, F. Wang, J. Zhang, and W. Ren, "Remaining useful life prediction of ultrasonic motor based on Elman neural network with improved particle swarm optimization," *Measurement,* vol. 143, pp. 27-38, 2019. https://doi.org/10.1016/j.measurement.2019.05.013.

[40] P. Jayapriya and S. Hemalatha, "Bacterial Foraging Optimization based Recurrent Neural Network Approach ForIdentification and Classification of Maize Plant Disease," in *2022 Third International Conference on Intelligent Computing Instrumentation and Control Technologies (ICICICT)*, 2022: IEEE, pp. 35-43. https://doi.org/10.1109/ICICICT54557.2022.9917607.