Received: February 13, 2023. Revised: March 15, 2023. 269

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Intrusion Detection Using Pareto Optimality Based Grasshopper Optimization Algorithm with Stacked Autoencoder in Cloud and IoT Networks

Hymavathi Kanakadurga Bella[1]* **Vasundra Sanjeevulu[2]**

[1]*Jawaharlal Nehru Technological University Anantapur, Ananthapuramu, India*
[2]*JNTUA College of Engineering, Ananthapuramu,*
Constituent college of Jawaharlal Nehru Technological University Anantapur, Ananthapuramu, India
* Corresponding author's Email: reddykanakadurga3@gmail.com

Abstract: Currently, cloud computing and its application is a popular area of research in which, intrusion detection has become an imperative system for detecting several security breaches. The main motivation of this research is to develop an automated intrusion detection system (IDS) for the detection of intrusions in the cloud and internet of things (IoT) systems. After the acquisition of intrusion data from the NSL-KDD, CICIDS2017, Kyoto 2006+, and UNSW-NB15 databases, the Min-Max normalization approach is employed for data rescaling. Then, the relevant attributes/features are selected by proposing pareto optimality based grasshopper optimization algorithm (POGOA), where the selection of relevant features efficiently reduces the system's complexity and computational time. In the POGOA, the relevant features are selected based on pareto optimal solutions that help in enhancing the premature convergence and distribution rate of the traditional GOA. Further, the selected features are given to the stacked autoencoder model for classifying the normal and attack classes. The proposed POGOA with stacked autoencoder model's experiment is conducted using the Matlab environment. The proposed model shows better performance by means of precision, f1-score, accuracy, and recall when related to the comparative models. The proposed POGOA with stacked autoencoder model has obtained 99.32%, 99.84%, 99.56%, and 97.24% of detection accuracy on the CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15 databases.

Keywords: Cloud computing, Grasshopper optimization algorithm, Intrusion detection, Machine learning, Stacked autoencoder.

1. Introduction

Cloud computing is a recent development in information technology (IT), which provides facilities, a platform, and software as internet services [1]. Cloud computing represents the realization of an old ideal termed "computing for use", and it is slowly being adopted by businesses as private, public, and hybrid clouds. Its primary goal is to provide on-demand software and infrastructure services to consumers [2]. Cloud computing is an effective IT infrastructure platform, but it still requires considerable security efforts, to reduce its flaws. Because cloud data centers have a large quantity of personal and business information, security concerns and vulnerabilities related to cloud computing must be detected and mitigated [3]. The statistics show that the virtual network layer of cloud computing has seen an enormous number of security breaches in recent years [4]. To uncover assaults on cloud infrastructure, several IDS tools and approaches are now accessible on the cloud platform. Due to high traffic and its dynamic behavior, traditional or current IDS tools are proven to be useless and inefficient.

The IDS is often used in conjunction with a firewall to create a comprehensive security solution. These factors have compelled researchers to devise a new strategy for enhancing cloud computing's IDS. The ability to monitor network traffic and detect network intrusions is provided by the software [5]. The most common method for discovering assaults on cloud infrastructure is through the deployment of IDS. Because of the virtualized and dispersed nature

of cloud settings, it is difficult to create effective IDS. Attacks on systems and applications by cybercriminals lead to system and application failures, which is why cyber-attack protection is so important. When an assault is detected, the IDS stops it by discarding packets, resetting the connection, or blocking traffic [6]. The IDS is considered effective if it optimizes detection accuracy while minimizing the number of false alarms it generates. There are four main types of IDS: virtual machine-based (VMIDS), host-based (HIDS), network-based (NIDS), and perimeter-based (PIDS) [7]. The NIDS keeps track of, and analyses all network traffic, including inbound and outbound. For monitoring and alerting purposes, a HIDS is put in place on the hosts. When an intrusion attempt is made on a perimeter fence around a critical system infrastructure, such as the main server, the PIDS identifies and locates it. The VMIDS is identical to any of the three IDSs listed above, except it is run on a remote virtual machine instead of locally [8]. There is a low probability of false alarms with these sorts of systems, but the systems are not able to identify assaults that have not been previously recognized with no historical information [9-10]. To address this concern, a new machine learning-based framework is developed in this paper. The objectives of this paper are illustrated below:

- In the initial phase, the efficacy of the proposed machine learning based framework is validated on the cloud databases such as CICIDS2017, NSL-KDD, Kyoto 2006+ and UNSW-NB15, which are widely utilized databases in intrusion classification.
- Implemented min-max normalization approach and POGOA for rescaling and selecting the relevant attributes from the CICIDS2017, NSL-KDD, Kyoto 2006+ and UNSW-NB15 databases for better intrusion classification. The elimination of irrelevant attributes significantly diminishes the proposed framework's complexity and time. In the proposed POGOA, the target (relevant attributes) is chosen based on the Pareto optimal solutions that improve the premature convergence and distribution rate of the conventional GOA.
- By using the selected attributes, the stacked autoencoder model classifies the normal and attack intrusions in the cloud systems. The proposed framework's effectiveness is investigated using the evaluation measures like precision, f1-score, accuracy, and recall.

The manuscripts related to intrusion detection are reviewed in section 2. The undertaken methodology details, simulation outcomes, and the conclusion of the proposed framework are analyzed in sections 3, 4, and 5, respectively.

2. Related works

Bhushan and Gupta [11] performed a high-level attribute extraction based on a stacked autoencoder model. Application-level distributed denial-of-service (DDoS) attack detection using encrypted protocols, made use of the anomaly detection approach based on estimated statistics from network packets. Client-server negotiations were organized, and real user behavior was projected in the way it was supplied. A stacked autoencoder was used for evaluating the distribution of negotiations among these groups. Anomalies were client negotiations that diverged from the normal. Deep learning was used in the IoT fog environment to identify distributed threats. Instead of using shallow neural networks, the distributed detection approach was more efficient. A deep learning-based defense system for DDoS attack detection was envisaged in the software-defined networking (SDN) setup. The developed system was able to recognize patterns in the network traffic and track an attack's progress over time, but it was computationally complex.

Pillutla and Arjunan [12] used fuzzy self-organizing maps for cleaning up traffic after a DDoS assault on SDN. Although there were several IDS models available at the network layer, the existing models had obtained low detection accuracy and needed extensive computing resources. Furthermore, the IDS developed with the traditional networks cannot be used in the cloud platforms; as a result, a superior IDS model for the cloud was required.

Almiani [13] implemented a novel automated intrusion detection model for fog computing security against cyber-attacks. A new multi-layered recurrent neural network (RNN) model was implemented for fog computing security, which was close to the IoT devices and end-users. A balanced database: NSL-KDD was utilized for demonstrating the efficacy of the multi-layered RNN model, and its performance was validated using Cohen's kappa coefficients and Matthew's correlation coefficients. The simulation and the experimental results proved that the implemented model was robust and stable in intrusion detection. The implemented multi-layered RNN model was computationally expensive, where it required enormous amounts of data for obtaining better intrusion detection performance.

Benmessahel [14] integrated locust swarm optimization (LSO) with feed-forward neural

network (FNN) for effective intrusion attack classification. In this literature, the developed LSO-FNN model was employed in a series of experiments for studying its performance and capability in intrusion detection. The benchmark database: NSL-KDD was used for analyzing the developed LSO-FNN model's performance. The experimental investigation showed that the developed LSO-FNN model not only obtained better convergence speed but also achieved better reliability in intrusion detection. However, the LSO was easily trapped into local minima, which was a major problem in this literature.

Choobdar [15] incorporated stacked auto encoders for learning the features from the CICIDS2017 and NSL-KDD databases. In the next phase, the softmax classifier was utilized for training the system, and further, the system parameters were optimized in the final phase. The extensive experimental examination states that the above model had achieved efficient performance in the intrusion attack classification on the CICIDS2017 and NSL-KDD databases. The overfitting and the vanishing gradient were the major concerns associated with the developed model, which needs to be concentrated as a future extension.

Han [16] developed an effective automated intrusion detection system based on proximal policy optimization (PPO) for protecting the IoT environments against cyber-attackers. At first, the discriminative features were obtained from the acquired database by implementing a deep neural network (DNN), and then the similar extracted features were clustered by using the k-means clustering technique. Further, feature dimensionality reduction and intrusion classification were performed by utilizing PPO and fine-tuned neural networks. The effectiveness of the developed model was analyzed utilizing the CICIDS2017 database by means of f1-score, and the obtained results state that the developed model was computationally expensive.

Chikkalwar and Garapati [17] integrated autoencoder, support vector machine (SVM), and GOA for effective detection of network intrusions. The presented model effectively resolves overfitting and data imbalance issues in the intrusion detection, but it has an issue of high computational time, Aziz and Alfoudi [18] developed a new network intrusion detection model named restoration particle swarm optimization (RPSO) for selecting relevant features from the NSL-KDD database. Sandhya and Kumarappan [19] used spider monkey optimization algorithm, and Krishna and Arunkumar [20] implemented a hybrid model: PSO and gray wolf optimization (GWO) for IoT based network intrusion detection. However, the traditional optimization algorithms include optimization issues like poor premature convergence and distribution rate. To address the aforementioned problems, a novel POGOA with stacked autoencoder is introduced in this paper.

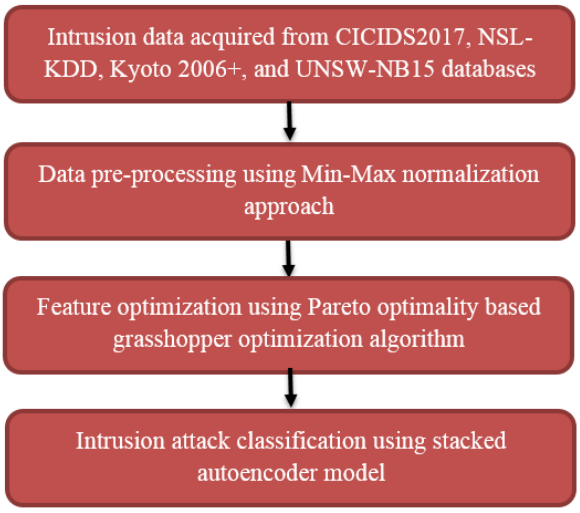

Figure. 1 Flow-chart of the proposed framework

3. Methods

For an effective intrusion attack classification, the proposed framework comprises four steps: Database description: CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15, pre-processing: Min-max normalization approach, feature optimization: POGOA and intrusion attack classification: Stacked autoencoder. The flowchart of the proposed framework is depicted in Fig. 1.

3.1 Database description and pre-processing

The POGOA with stacked autoencoder model's performance is evaluated on four online databases such as CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15. In recent periods, the CICIDS2017 is one of the advanced intrusion detection databases, which almost covers up to data attacks like infiltration attacks, web attacks, DoS attacks, DDoS attacks, heart-bleed attacks, brute force attacks, and Botnet. The CICIDS2017 database includes 225,745 packages, while every package has 80 features with labels [21]. The NSL-KDD is an updated version of the KDD'99 database, which is a benchmark database that helps researchers in comparing the proposed model with the existing intrusion detection models. The NSL-KDD test set includes 311,027 original records and 77,289 distinct records in comma separated values (CSV) format, and the train set has 4,898,431 original records and 1,074,992 distinct

records in CSV format [22]. In addition, the Kyoto 2006+ database has 24 features, 113,120 normal data, 118,191 attack data, and 231,311 number of sample sizes. The UNSW-NB15 database has nine attacks like worms, shellcode, reconnaissance, generic, exploits, DoS, backdoors, analysis, and fuzzers. The UNSW-NB15 database has 49 features, 82,332 testing records, and 175,341 training records.

After the acquisition of CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15 databases, the Min-Max normalization approach is employed for data rescaling that preserves the shape of the original distribution. The Min-Max normalization approach scales the data features in the range of [-1, 1] or [0, 1], if there are negative feature values in the database. After denoising, the acquired data, the relevant features/attributes are selected by implementing a modified version of GOA, where this process helps in the reduction of computational complexity and time.

3.2 Feature optimization

After data pre-processing, the relevant attributes are selected by implementing POGOA. The conventional GOA is an effective swarm intelligence optimization algorithm, which is inspired by the swarming and foraging behaviors of grasshoppers [23, 24]. The conventional GOA is successfully employed in solving several optimization issues [25, 26] and the i^{th} grasshopper's position is represented asVi, it is mathematically denoted in Eq. (1).

$$Vi = Si + Oi + Ai \tag{1}$$

Where, Ai indicates wind advection, Oi represents gravitational force and Si indicates social interactions experienced by i^{th} grasshoppers. In the conventional GOA, these three elements replicate the grasshopper's motions that are defined in Eqs. (2), (3), and (4). [27, 28].

$$Ai = uew \tag{2}$$

$$Oi = -geg \tag{3}$$

$$Si = \sum_{j=1, j\neq i}^{N} s(d_{ij})\grave{d}_{ij} \tag{4}$$

Where, u represents constant drift, $ew\ and\ eg$ indicate unity vectors, which are directed towards the global centers, $s(r) = \frac{ke-r}{l-e-r}$ indicates the strength of the social forces, $\grave{d}_{ij} = \frac{vj-vi}{d_{ij}}$ and $d_{ij} = |vj - vi|$ represents a unit vector, l represents the attractive length scale, k indicates attraction intensity, and o specifies gravitational constant. The grasshopper's nymph motion Vi is computed utilizing the elements Oi, Ai and Si, and it is mathematically mentioned in Eq. (5).

$$Vi = j = 1, jiN\ s\big(d_{ij}\big)(d_{ij}) - geg + uew \tag{5}$$

Where, N indicates the number of grasshoppers. The GOA performs exploitation and exploration for identifying the global optimal approximations to resolve the optimization problems. Eq. (5) is updated as mentioned in Eq. (6).

$$Vid = cj = 1, jiNcubd - Ibd2s(|vjd - vid|)vj - vid_{ij} + Td \tag{6}$$

Where, ubd denotes the upper bound, Ibd specifies the lower bound, Td represents the target value, and c indicates decreasing coefficients, which are inversely proportional to the iteration numbers, as determined in Eq. (7).

$$C = cmax - Icmax - cmin \times Maxitr \tag{7}$$

Where, $Maxitr = 100$ represents a maximum number of iterations, $cmin$ denotes the minimum c value, I denotes the present iteration number, and $cmax$ represents maximum c value. In the traditional GOA, the target values are chosen based on the best solutions obtained so far, which are ineffective in the larger intrusion databases. Therefore, the target values are selected based on the Pareto optimal solutions in the POGOA that superiorly improves the premature convergence and distribution rates. The neighborhood solutions are considered and counted as the quantitative measures in the Pareto optimal solutions for determining the crowded-ness regions.

The probability of the target value selection Pr_i is defined in Eq. (8). In the proposed POGOA, a roulette wheel approach is applied along with Pr_i for choosing the target values from the archives.

$$Pr_i = \frac{1}{N_i} \tag{8}$$

Where, N_i represents the number of solutions. The assumed parameters of the proposed POGOA are determined as follows: upper bound is 11, population size is 22, minimum c value is 0.2, lower bound is 0, and maximum c value is 1. The selected 28 attributes of CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15 databases are given to the stacked autoencoder for intrusion attack classification.

3.3 Intrusion attack classification

After selecting the optimal features from the CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15 databases, the stacked autoencoder classifier is employed for classifying the intrusion attacks. The stacked autoencoder classifier comprises multi-layer autoencoders that obtain higher-level representation of the selected features by reconstructing its structure. In the input layer, the original information is encoded for obtaining the higher-level features of the middle hidden layers, and further, the respective feature information is reconstructed by decoding it. By minimizing the reconstruction error, the stacked autoencoder networks are trained. The original training data is considered as $x^{(i)}$ and the hidden layer $y^{(i)}$ is mathematically expressed in Eq. (9) [29].

$$y^{(i)} = f(W_1^T \times x^{(i)} + b) \tag{9}$$

Where f indicates encoding function (tangent activation function), W_1^T represents the weight matrix of the encoder, and b indicates bias vector. Further, the output $z^{(i)}$ is obtained by decoding the original information, which is mathematically represented in Eq. (10). Then, the objective $J(X, Z)$ is minimized for training the autoencoder that is defined in Eq. (11).

$$z^{(i)} = g(W_2^T \times y^{(i)} + b) \tag{10}$$

Where, g denotes the decoding function, W_2^T indicates the weight matrix of the decoder and b indicates the bias vector [30, 31].

$$J(X, Z) = \frac{1}{2}\sum_{i=1}^{M} \left\| x^{(i)} - z^{(i)} \right\|^2 \tag{11}$$

Where, M indicates the number of autoencoder. Generally, the stacked autoencoder classifier is trained based on a layer-by-layer greedy model. Here, the hidden layer vectors of the upper layer are used as the input of the successive layers, and this step is named pre-training. Further, the pre-trained network weights are connected and then the weights of the final network are obtained by fine-tuning. The assumed parameters of the stacked autoencoder classifier are stated as follows: maximum iterations: softmax learning is 100, sparsity proportion is 0.15, sparsity regularization is 4, L2 weight regularization is 0.004, and the number of hidden layers is 100. The obtained experimental outcomes of the POGOA with stacked autoencoder model are mentioned in the upcoming section.

4. Experimental results

The proposed POGOA with stacked autoencoder model's performance is validated using Matlab R2020a software tool with Intel core i3 processor, 8 GB Random Access Memory and 4TB hard disk. In this manuscript, the proposed POGOA with stacked autoencoder model's performance is analyzed by using the performance measures like precision, f1-score, accuracy and recall, and it is mathematically specified in Eqs. (12), (13), (14), and (15). In this scenario, the false negatives (FN) values specify that the attack classes are inaccurately determined as normal classes, and the false positives (FP) values indicate that the attack classes are precisely identified as attack classes. Additionally, the true positives (TP) values indicate that the normal classes are accurately identified as normal classes, and the true negatives (TN) values denote that the normal classes are inaccurately determined as attack classes. The formulae to compute precision, f1-score, accuracy, and recall are denoted in Eqs. (12), (13), (14), and (15).

$$Precision = \frac{TP}{TP+FP} \times 100 \tag{12}$$

$$F1 - score = 2\frac{precision \times recall}{precision + recall} \times 100 \tag{13}$$

$$Accuracy = \frac{TN+TP}{TP+TN+FP+FN} \times 100 \tag{14}$$

$$Recall = \frac{TP}{TP+FN} \times 100 \tag{15}$$

4.1 Analysis of the CICIDS2017 database

The proposed POGOA with stacked autoencoder model's efficacy is analyzed on the CICIDS2017 database with fivefold cross-validation (80:20% training and testing of data). As specified in Table 1, an ablation study is performed with different optimizers: Squirrel optimization algorithm (SOA), GOA and POGOA, and classifiers: RNN, long short term memory (LSTM) network and stacked autoencoder. By inspecting the obtained results, the combination of POGOA with stacked autoencoder has higher classification results on the CICIDS2017 database with precision of 99.58%, f1-score of 99.18%, accuracy of 99.32%, and recall of 99.50% on the CICIDS2017 database. On the other hand, the combination of POGOA with a stacked autoencoder consumed a computational time of 44.22 seconds, which is limited compared to other combinations. The graphical diagram of the experimental results obtained by the proposed POGOA with stacked

Table 1. Obtained results of the proposed POGOA with stacked autoencoder model on the CICIDS2017 database

Classifiers	Optimizers	Precision (%)	F1-score (%)	Accuracy (%)	Recall (%)
RNN	SOA	90.32	92	91.20	88.92
	GOA	91.20	92.69	93.44	93.28
	POGOA	93.28	93.20	92.92	94.47
LSTM	SOA	93.12	92.10	93.28	92.22
	GOA	94.77	94.36	95.68	94.38
	POGOA	95.44	95.38	95.90	95
Stacked autoencoder	SOA	96.80	96.55	96.49	96.06
	GOA	97.02	97.35	97.40	96.57
	POGOA	99.58	99.18	99.32	99.50

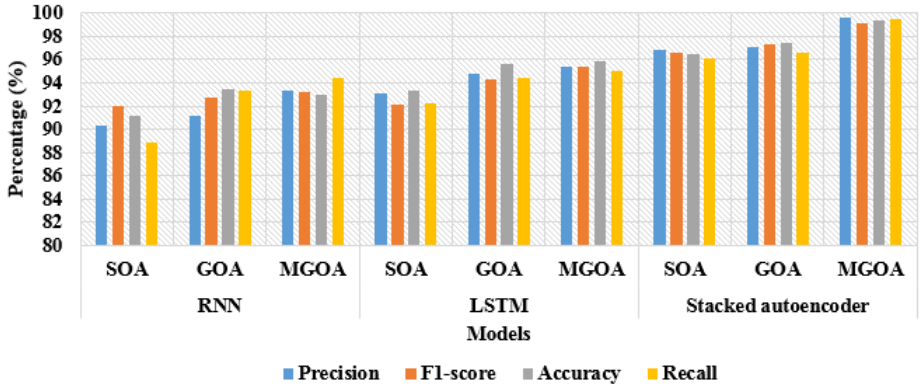

Figure. 2 Graphical diagram of the experimental results obtained by the proposed model on the CICIDS2017 database

Table 2. Obtained results of the proposed POGOA with stacked autoencoder model on the NSL-KDD database

Classifiers	Optimizers	Precision (%)	F1-score (%)	Accuracy (%)	Recall (%)
RNN	SOA	89.40	90.45	88.22	88.34
	GOA	90.28	91.60	90.54	90.90
	POGOA	92.12	92.28	92.80	92
LSTM	SOA	90.78	93.67	92.77	92.20
	GOA	92.60	92.30	93.60	93.30
	POGOA	93.42	94.44	94.62	95.88
Stacked autoencoder	SOA	95.84	95.50	95.90	96.84
	GOA	96.67	96.88	96.88	96.90
	POGOA	99.67	99.72	99.84	99.52

autoencoder model on the CICIDS2017 database is specified in Fig. 2.

4.2 Analysis of the NSL-KDD database

In this section, the proposed POGOA with stacked autoencoder model's effectiveness is tested on the NSL-KDD database by means of precision, f1-score, accuracy and recall, and the results are specified in Table 2. Similar to the CICIDS2017 database, the combination: POGOA with stacked autoencoder has achieved higher classification results on the NSL-KDD database with precision of 99.67%, f1-score of 99.72%, accuracy of 99.84%, and recall of 99.52%. In the NSL-KDD database, the combination: POGOA with stacked autoencoder consumed a computational time of 49.34 seconds, which is better related to other combinations, due to the selection of the relevant attributes by POGOA. The graphical diagram of the experimental results obtained by the proposed model on the NSL-KDD database is represented in Fig. 3.

4.3 Analysis of the Kyoto 2006+ and UNSW-NB15 databases

Similar to the previous two databases, the proposed POGOA with stacked autoencoder model obtained higher classification results related to other comparative models on the Kyoto 2006+ and UNSW-NB15 databases. The proposed POGOA with stacked autoencoder model has obtained 99.56%, and 97.24% of detection accuracy on the Kyoto 2006+, and UNSW-NB15 databases. Compared to the existing

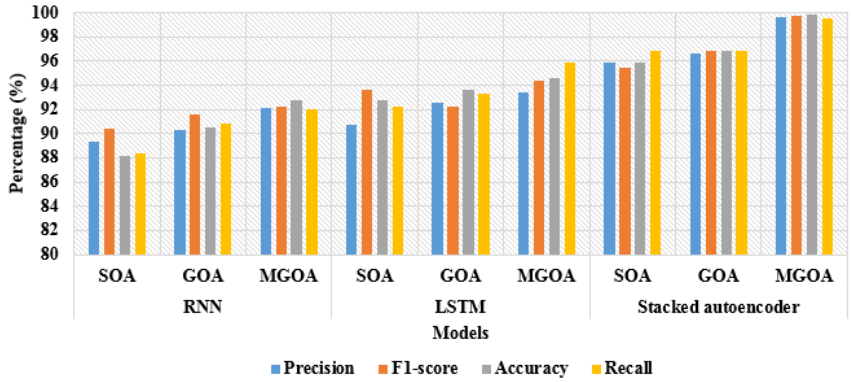

Figure. 3 Graphical diagram of the experimental results obtained by the proposed model on the NSL-KDD database

Table 3. Obtained results of the proposed POGOA with stacked autoencoder model on the Kyoto 2006+ database

Classifiers	Optimizers	Precision (%)	F1-score (%)	Accuracy (%)	Recall (%)
RNN	SOA	90.48	91.40	90.29	90.36
	GOA	91.28	92.69	91.58	92.97
	POGOA	93.17	93.28	93.88	92.67
LSTM	SOA	91.76	94.68	94.76	94.24
	GOA	93.65	95.33	95.66	94.34
	POGOA	94.44	94.44	96.60	95.90
Stacked autoencoder	SOA	96.85	96.54	97.09	97.81
	GOA	97.60	97.84	97.87	98.93
	POGOA	99.12	99.23	99.56	99.24

Table 4. Obtained results of the proposed POGOA with stacked autoencoder model on the UNSW-NB15 database

Classifiers	Optimizers	Precision (%)	F1-score (%)	Accuracy (%)	Recall (%)
RNN	SOA	90.43	91.11	90.43	90.22
	GOA	91.21	92.35	92.59	92.82
	POGOA	93.19	93.78	94.84	93.22
LSTM	SOA	91.33	94.34	94.90	93.42
	GOA	93.68	94.33	95.64	94.82
	POGOA	94.56	95.22	96.43	96.80
Stacked autoencoder	SOA	96.22	96.58	97	97.68
	GOA	97.33	97.90	97.09	97.92
	POGOA	98.80	98.92	97.24	98.58

classification techniques, the stacked autoencoder greatly decreases the input data noise, which makes this deep-learning model more effective in intrusion classification. In the Kyoto 2006+ and UNSW-NB15 databases, the combination: POGOA with stacked autoencoder consumed a limited computational time of 29.30 and 44.02 seconds, and it is better than the comparative models. The graphical diagram of the experimental results obtained by the proposed model on the Kyoto 2006+ and UNSW-NB15 database is presented in Figs. 4 and 5.

4.4 Comparative analysis

The comparative analysis between the proposed POGOA with the stacked autoencoder model and the prior models is mentioned in Tables 5 and 6. Almiani [13] implemented a new intrusion detection model: multi-layered RNN for improving fog computing security. The developed multi-layered RNN model's performance was analyzed on the NSL-KDD database, where it achieved 92.18% of detection accuracy. Benmessahel [14] combined LSO with FNN for effective intrusion detection. The experimental analysis showed that the presented LSO-FNN model achieved 94.02% of detection accuracy on the NSL-KDD database. Choobdar [15] incorporated stacked autoencoder with softmax classifiers for intrusion attack classification on the CICIDS2017 database. The developed model has averagely obtained 98.50% of detection accuracy on the CICIDS2017 database. Chikkalwar and Garapati [17] integrated autoencoder, SVM and GOA for

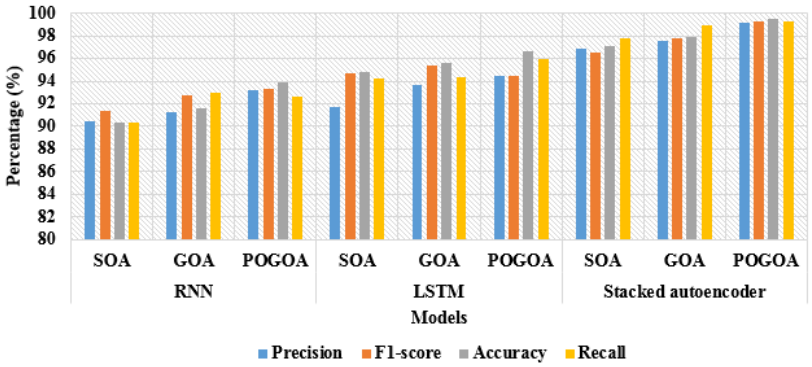

Figure. 4 Graphical diagram of the experimental results obtained by the proposed model on the Kyoto 2006+ database

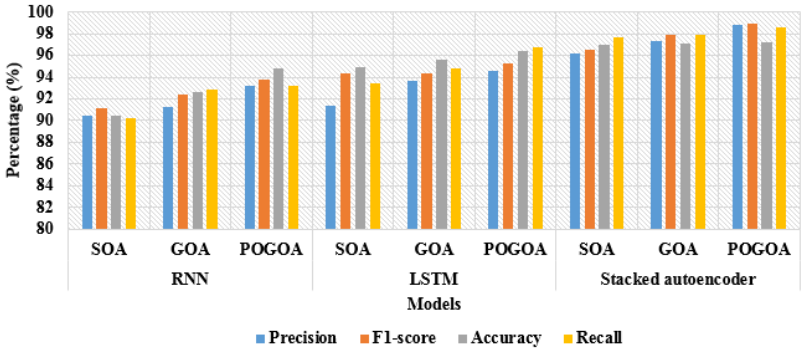

Figure. 5 Graphical diagram of the experimental results obtained by the proposed model on the UNSW-NB15 database

Table 5. Comparative results by means of detection accuracy

Models	Database	Detection accuracy (%)
Multi-layered RNN [13]	NSL-KDD	92.18
LSO with FNN [14]	NSL-KDD	94.02
Stacked autoencoder [15]	CICIDS2017	98.50
POGOA with stacked autoencoder	NSL-KDD	98.12
	CICIDS2017	98.74

Table 6. Comparative results by means of detection accuracy, precision, f1-score, recall, and computational time

Models	Database	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Time (seconds)
Autoencoder-SVM-GOA [17]	UNSW-NB15	95.30	94.10	95.20	94.65	41
	CICIDS2017	99.20	99.10	99.10	99.10	36
	NSL-KDD	99.60	99.50	99.40	99.45	27
	Kyoto 2006+	99.10	99.10	99.10	99.10	16
POGOA with stacked autoencoder	UNSW-NB15	97.24	98.80	98.58	98.92	32.12
	CICIDS2017	99.32	99.58	99.50	99.18	27.30
	NSL-KDD	99.84	99.67	99.52	99.72	18
	Kyoto 2006+	99.56	99.12	99.24	99.23	9.48

effective detection of network intrusions. Whereas, the presented model has obtained higher intrusion classification outcomes by means of precision, f1-score, accuracy, and recall on the CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15 databases.

As specified in Tables 5 and 6, the proposed POGOA with stacked autoencoder model obtained higher detection accuracy compared to the existing research papers. In addition to this, the selection of optimal attributes by the POGOA significantly reduces the system complexity to linear and computational time, which are the major concerns highlighted in the literature section. The proposed POGOA with stacked autoencoder model consumed limited computational time of 32.12, 27.30, 18, and 9.48 seconds on the UNSW-NB15, CICIDS2017, NSL-KDD, and Kyoto 2006+ databases.

5. Conclusion

In this paper, a POGOA with stacked autoencoder model is implemented for effective intrusion detection in the cloud and IoT systems. The proposed framework is implemented in the Matlab environment on the CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15 databases. The proposed framework comprises two major phases: feature optimization using POGOA and intrusion classification by stacked autoencoder. As mentioned in the comparative analysis section, the proposed POGOA with stacked autoencoder model has achieved higher classification results compared to the existing models: stacked autoencoder, LSO with FNN, multi-layered RNN, and autoencoder-SVM-GOA. The POGOA with stacked autoencoder model achieved 99.32%, 99.84%, 99.56%, and 97.24% of detection accuracy on the CICIDS2017, NSL-KDD, Kyoto 2006+, and UNSW-NB15 databases. Still, the proposed model needs to concentrate on the data imbalance problem to further enhance its detection accuracy. Therefore, as a future extension, an effective data balancing technique is incorporated with the proposed POGOA with stacked autoencoder model to further achieve better results in intrusion attack classification.

Notations

Parameter	Definition
Ai	Wind advection
Oi	Gravitational force
Si	Social interactions experienced by i^{th} grasshoppers
u	Constant drift
$ew\ and\ eg$	Unity vectors
$s(r)$	Strength of the social forces
l	Attractive length scale
k	Attraction intensity
o	Gravitational constant
Vi	Grasshopper's nymph motion
ubd	Upper bound
Ibd	Lower bound
Td	Target value
c	Decreasing coefficients
$Maxitr$	Maximum number of iterations
$cmin$	Minimum c value
I	Present iteration number
$cmax$	Maximum c value
Pr_i	Probability of the target value selection
N_i	Number of solutions
W_1^T	Weight matrix of the encoder
f	Encoding function
b	Bias vector
W_2^T	Weight matrix of the decoder
g	Decoding function
M	Number of autoencoder

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

The paper conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, visualization, have been done by 1st author. The supervision and project administration, have been done by 2nd author.

References

[1] A. P. Achilleos, K. Kritikos, A. Rossini, G. M. Kapitsaki, J. Domaschka, M. Orzechowski, D. Seybold, F. Griesinger, N. Nikolov, D. Romero, and G. A. Papadopoulos, "The cloud application modelling and execution language", *Journal of Cloud computing*, Vol. 8, p. 20, 2019.

[2] P. P. Kumar, P. S. Kumar, and P. J. A. Alphonse, "Attribute based encryption in cloud computing: A survey, gap analysis, and future directions", *Journal of Network and Computer Applications*, Vol. 108, pp. 37-52, 2018.

[3] T. Halabi and M. Bellaiche, "Towards quantification and evaluation of security of Cloud Service Providers", *Journal of Information Security and Applications*, Vol. 33, pp. 55-65, 2017.

[4] T. Halabi and M. Bellaiche, "A broker-based framework for standardization and management of Cloud Security-SLAs", *Computers & Security*, Vol. 75, pp. 59-71, 2018.

[5] M. R. Chinnaiah and N. Niranjan, "Fault tolerant software systems using software configurations for cloud computing", *Journal of Cloud Computing*, Vol. 7, pp. 3, 2018.

[6] R. Kumar and R. Goyal, "On cloud security requirements, threats, vulnerabilities and countermeasures: A survey", *Computer Science Review*, Vol. 33, pp. 1-48, 2019.

[7] Y. Li, R. Ma, and R. Jiao, "A hybrid malicious code detection method based on deep learning", *International Journal of Security and Its Applications*, Vol. 9, No. 5, pp. 205-216, 2015.

[8] T. Ma, F. Wang, J. Cheng, Y. Yu, and X. Chen,

"A Hybrid Spectral Clustering and Deep Neural Network Ensemble Algorithm for Intrusion Detection in Sensor Networks", *Sensors*, Vol. 16, No. 10, p. 1701, 2016.

[9] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An Adaptive Ensemble Machine Learning Model for Intrusion Detection", *IEEE Access*, Vol. 7, pp. 82512-82521, 2019.

[10] K. Jiang, W. Wang, A. Wang, and H. Wu, "Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network", *IEEE Access*, Vol. 8, pp. 32464-32476, 2020.

[11] K. Bhushan and B. B. Gupta, "Distributed denial of service (DDoS) attack mitigation in software defined network (SDN)-based cloud computing environment", *Journal of Ambient Intelligence and Humanized Computing*, Vol. 10, No. 5, pp. 1985-1997, 2019.

[12] H. Pillutla and A. Arjunan, "Fuzzy self organizing maps-based DDoS mitigation mechanism for software defined networking in cloud computing", *Journal of Ambient Intelligence and Humanized Computing*, Vol. 10, No. 4, pp. 1547-1559, 2019.

[13] M. Almiani, A. A. Ghazleh, A. A. Rahayfeh, S. Atiewi, and A. Razaque, "Deep recurrent neural network for IoT intrusion detection system", *Simulation Modelling Practice and Theory*, Vol. 101, p. 102031, 2020.

[14] I. Benmessahel, K. Xie, M. Chellal, and T. Semong, "A new evolutionary neural networks based on intrusion detection systems using locust swarm optimization", *Evolutionary Intelligence*, Vol. 12, No. 2, pp. 131-146, 2019.

[15] P. Choobdar, M. Naderan, and M. Naderan, "Detection and Multi-Class Classification of Intrusion in Software Defined Networks Using Stacked Auto-Encoders and CICIDS2017 Dataset", *Wireless Personal Communications*, Vol. 123, No. 1, pp. 437-471, 2022.

[16] H. Han, H. Kim, and Y. Kim, "Correlation between Deep Neural Network Hidden Layer and Intrusion Detection Performance in IoT Intrusion Detection System", *Symmetry*, Vol. 14, No. 10, p. 2077, 2022.

[17] S. R. Chikkalwar and Y. Garapati, "Autoencoder-support vector machine-grasshopper optimization for intrusion detection system", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 4, pp. 406-414, 2020, doi: 10.22266/ijies2022.0831.36.

[18] M. R. Aziz and A. S. Alfoudi, "Feature Selection of The Anomaly Network Intrusion Detection Based on Restoration Particle Swarm Optimization", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 5, pp. 592-600, 2022, doi: 10.22266/ijies2022.1031.51.

[19] E. Sandhya and A. Kumarappan, "Enhancing the Performance of an Intrusion Detection System Using Spider Monkey Optimization in IoT", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 6, pp. 30-39, 2021, doi: 10.22266/ijies2021.1231.04.

[20] E. S. P. Krishna and T. Arunkumar, "Hybrid particle swarm and gray wolf optimization algorithm for IoT intrusion detection system", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 4, pp. 66-76, 2021, doi: 10.22266/ijies2021.0831.07.

[21] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems", *International Journal of Engineering & Technology*, Vol. 7, No. 3, pp. 479-482, 2018.

[22] G. Meena and R. R. Choudhary, "A review paper on IDS classification using KDD 99 and NSL KDD dataset in WEKA", In: *Proc. of International Conference on Computer, Communications and Electronics (Comptelix)*, Jaipur, India, pp. 553-558, 2017.

[23] S. Z. Mirjalili, S. Mirjalili, S. Saremi, H. Faris, and I. Aljarah, "Grasshopper optimization algorithm for multi-objective optimization problems", *Applied Intelligence*, Vol. 48, No. 4, pp. 805-820, 2018.

[24] V. Ravuri and S. Vasundra, "Moth-flame optimization-bat optimization: Map-reduce framework for big data clustering using the Moth-flame bat optimization and sparse Fuzzy C-means", *Big Data*, Vol. 8, No. 3, pp. 203-217, 2020.

[25] S. Vasundra and G. Rajeswarappa, "Hybrid Grasshopper and Improved Bat Optimization Algorithms-based clustering scheme for maximizing lifetime of Wireless Sensor Networks (WSNs)", *International Journal of Intelligent Engineering and Systems*, Vol. 15, pp. 536-546, 2022, doi: 10.22266/ijies2022.0630.45.

[26] S. Vasundra and A. Balaram, "A Hybrid Soft Computing Technique for Software Fault Prediction based on Optimal Feature Extraction and Classification", *International Journal of Computer Science and Network Security*, Vol. 22, No. 5, pp. 348-358, 2022.

[27] Y. Meraihi, A. B. Gabis, S. Mirjalili, and A. R. Cherif, "Grasshopper Optimization Algorithm: Theory, Variants, and Applications", *IEEE Access*, Vol. 9, pp. 50001-50024, 2021.

[28] H. Hichem, M. Elkamel, M. Rafik, M. T.

Mesaaoud, and C. Ouahiba, “A new binary grasshopper optimization algorithm for feature selection problem”, *Journal of King Saud University-Computer and Information Sciences*, Vol. 34, No. 2, pp. 316-328, 2019.

[29] M. Yu, T. Quan, Q. Peng, X. Yu, and L. Liu, “A model-based collaborate filtering algorithm based on stacked AutoEncoder”, *Neural Computing and Applications*, Vol. 34, No. 4, pp. 2503-2511, 2022.

[30] S. Zhou, Z. Xue, and P. Du, “Semisupervised Stacked Autoencoder With Cotraining for Hyperspectral Image Classification”, *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 57, No. 6, pp. 3813-3826, 2019.

[31] S. Tao, T. Zhang, J. Yang, X. Wang, and W. Lu, "Bearing fault diagnosis method based on stacked autoencoder and softmax regression", In: *Proc. of 34th Chinese Control Conference (CCC)*, Hangzhou, China, pp. 6331-6335, 2015.