ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20250225002421
Original scientific paper

Optimized Intrusion Detection in IoT Networks Using Ensemble Deep Learning and Ant Colony Optimization

R. PANNEERSELVI*, J. VISUMATHI

Abstract: Many innovative applications in home automation, industry, health and environmental monitoring are possible because of the Internet of Things (IoT). At the same time, the rise in devices can make cybersecurity attacks more likely. Intrusion Detection Systems (IDSs) play a vital role in protecting networks by identifying and responding to attacks. Because IoT networks are used more often, there are privacy concerns since IDS use big data sets for deep learning (DL) and machine learning (ML). It is important to use effective DL/ML-based IDS systems to spot and classify attacks on IoT networks. The paper suggests using Ensemble Deep Learning Models and Ant Colony Optimization for an improved Intrusion Detection System (OIDS-EDLMACO). The goal is to improve detection of attacks in IoT networks by applying advanced ensemble models. Min-max normalization is used to preprocess the data at the start. After that, the Chimp Optimization Algorithm (ChOA) is used to select the most important features. OIDS-EDLMACO uses an ensemble of Bi-LSTM, SNN and SAE models for classification. Also, Ant Colony Optimization (ACO) is applied to tune the hyperparameters of the ensemble classifier for better performance. The OIDS-EDLMACO model reached an average accuracy of 99.27%, precision of 98.17%, recall of 98.17%, F1-score of 98.17% and detection rate of 98.86%. The findings show that OIDS-EDLMACO is effective for detecting intrusions. Future research will concentrate on protecting data during training and making sure federated IoT systems can scale in real time.

Keywords: ant colony optimization; deep learning; ensemble learning; internet of things; intrusion detection system

1 INTRODUCTION

Internet of Things (IoT) gadgets are lightweight and lower-energy, and address their obtainable computation and energy to perform core application functionalities [1]. Cyber-attacks aiming at IoT devices are considered to rise the improvement of IoT. Multiple IoT devices are connected to internet, permitting an absence of security control misuse. Various security attacks are focused on IoT which contains diverse vulnerabilities [2]. As the IoT is disposed to many threats, it is important to classify the threats and proper susceptibilities to examine the IoT. During certain investigations, it is established that jamming, wormhole, routing, DoS, flooding, sinkhole, worm attacks, a man in the middle, and viruses possibly arise in IoT methods [3]. Flooding and Denial of Service (DoS) attack occurs in generating IoT platforms. Botnet threat is now progressively attaining popularity. AI is mainly employed to identify such IoT threats [4]. IDS is a monitoring device or software that holds track of the system to protect and flags the administrator if any suspicious actions are recognized [5]. It may identify the malicious threat and normal data that can be recognized with classical security mechanisms. The IDS is partitioned into dual classes per the recognition method called signature and anomaly-based recognition methods [6].

Investigators have improved IDSs utilizing novel detection methods like ML-based IDS that employs machine learning (ML) models to distinguish between attack and normal packets [7]. ML models can learn from huge datasets and comprehend the patterns and behaviors of data anomalies [8]. Nevertheless, complex algorithms and large datasets needed essential tools. Therefore, investigators should utilize feature selection models and data pre-processing to decrease the difficulty of ML models. However, decreasing complexity can undermine the precision of classification method [9]. Thus, utilizing a ML classification algorithms and Deep Learning (DL) data generator model may allow private and wide threat coverage to attain higher performance with lower model testing and training period to fit the lightweight gadgets endorsed by IoT systems [10].

This research proposes the OIDS-EDLMACO technique which is based on ensemble deep learning models and ant colony optimization. At the beginning, the data is processed by min-max normalization to make it more usable. In the process of classification, the OIDS-EDLMACO model creates ensemble models like Bi-LSTM, SNN and SAE. Using the ant colony optimization (ACO) algorithm, the process of choosing hyperparameters is carried out to improve the classification results of ensemble classifiers. The OIDS-EDLMACO algorithm can be evaluated using database.

2 LITERATURE REVIEW

The rise of IoT devices has led to more connectivity and automation which has changed both industries and daily routines. Modern society relies on IoT in homes, on our bodies, in factories and in important infrastructure. At the same time, having so many devices connected creates a major issue: a much larger area that cybercriminals can target. Because IoT devices are so diverse, have limited computing abilities and are sometimes not secured well, they are easy targets for those who want to do harm. Therefore, IoT networks need advanced and intelligent Intrusion Detection Systems (IDSs) to protect them from new and advanced threats. Because traditional security measures do not meet the needs of IoT, there is a need to find and develop better intrusion detection techniques. Intrusion Detection Systems are important for network security because they watch over network traffic and system actions for any suspicious activity. They are responsible for spotting and dealing with intrusions which helps to stop or lessen the damage caused by cyberattacks. In the world of IoT, IDSs have to deal with special difficulties.

Using machine learning and deep learning for intrusion detection in IoT networks brings up major privacy issues. Training these models usually needs a lot

of data and this data might include private information about users. It is very important to keep this data private and scientists are still working on ways to use machine learning without compromising privacy in intrusion detection. For example, federated learning trains models using data from different sources without sharing the data which could solve this issue.

This paper discusses how to develop an ideal intrusion detection system for IoT networks by using ensemble deep learning models and ant colony optimization. The new method, OIDS-EDLMACO, is designed to improve how accurately and efficiently attacks are detected by using several deep learning models and optimizing their parameters with a bio-inspired algorithm. The main idea in ensemble learning is to use several individual models together to get a stronger and more accurate result. Because deep learning models can spot complex patterns in big datasets, they are useful for intrusion detection. Yet, the success of deep learning models relies a lot on how hyperparameters such as the number of layers, the learning rate and the batch size are set. It can take a lot of time to adjust these hyperparameters by hand and the results may not be as good as expected. It is at this point that optimization algorithms such as ant colony optimization, are used.

The OIDS-EDLMACO method includes a number of separate stages. The first task is to preprocess the raw data from IoT. This generally requires cleaning the data, dealing with missing values and making the data fall within a similar range. Normalizing features helps each one contribute the same way and prevents those with larger values from overpowering the model. Many people use min-max normalization to solve this problem. Feature selection is done after the preprocessing stage. Feature selection is used to choose the most important features from the data which makes the data easier to handle and learn from. It also helps avoid overfitting which happens when the model fits the training data perfectly but fails on new data. The Chimp Optimization Algorithm (ChOA) is used in OIDS-EDLMACO to select important features. ChOA is designed like a chimpanzee's hunting behavior. It is capable of solving many optimization problems, for example, feature selection.

The main strength of the OIDS-EDLMACO method is its collection of deep learning models. Bi-directional Long Short-Term Memory (Bi-LSTM), Siamese Neural Network (SNN) and Stacked Autoencoder (SAE) are the three models in the ensemble. Bi-LSTMs are a kind of RNN that are specially designed for analyzing data that comes in a sequence such as network traffic. They are able to see connections between events that happened far apart in time which helps them find patterns that cover many time steps. SNNs are made to identify how similar different data points are. They are helpful in finding suspicious behavior since intrusions usually behave differently than regular traffic. SAEs are able to learn different levels of information from the data they process. They can find detailed patterns in the data which may help improve how accurately intrusions are detected. The ensemble combines the predictions of these three models, leveraging their diverse strengths to achieve higher accuracy than any individual model could achieve alone.

At the end, the Ant Colony Optimization (ACO) algorithm is applied to find the best values for the hyperparameters of the deep learning models and the ensemble. ACO is an optimization algorithm that works like ants while foraging for food. Ants leave pheromones on the ground to communicate with others. Other ants use the strongest pheromone trails to find the best places to get food. ACO searches the space of hyperparameters by testing different values and improving them until it finds the best combination. This step is very important for ensuring the best performance of the OIDS-EDLMACO method.

OIDS-EDLMACO is assessed using a commonly used intrusion detection dataset. The approach is tested against other advanced intrusion detection methods to show its effectiveness. The results of accuracy, precision, recall, F1-score and detection rate are carefully examined. It is clear from the experiments that OIDS-EDLMACO performs much better in detecting intrusions and is more efficient than other approaches. The use of ensemble learning, ChOA for feature selection and ACO for hyperparameter optimization makes OIDS-EDLMACO perform well. It demonstrates that using hybrid methods based on deep learning and optimization can help improve intrusion detection in the difficult context of IoT networks. More research could focus on using privacy-preserving methods to make IoT networks even more secure and private.

3 PROPOSED METHODOLOGY

This paper proposes an OIDS-EDLMACO technique. The main intention of OIDS-EDLMACO method is to improve the attack detection method in IoT networks using state-of-the-art ensemble models. The proposed model contains various processes such as data pre-processing, feature selection, classification, and parameter tuning. OIDS-EDLMACO is different in the fact that it uses ChOA for finding important features and ACO to adjust hyperparameters, plus a deep combination of BiLSTM, SNN and SAE. While traditional ensemble IDS use standard methods to choose features and adjust hyperparameters, OIDS-EDLMACO efficiently searches for the best features and configurations. In addition, including SNN in the ensemble improves the detection of unusual IoT data, as this type of data is not well represented in current ensemble models. Fig. 1 depicts the overall workflow of OIDS-EDLMACO model.

3.1 Data Pre-Processing

Min-max normalization is first used to turn the input data into a form that can be used advantageously. Min-max normalization is a way to prepare data for IDS by adjusting sensor data within a certain range [0, 1] [18]. It converts raw data while maintaining relationships among values, enhancing the performance of ML methods. This approach is important to handle heterogeneous sensor data with changing magnitudes and units. By normalizing attributes, it stops control of features with large scales and improves anomaly detection precision. It guarantees quicker convergence in training methods, resulting in more effective and accurate threat detection in IoT networks.

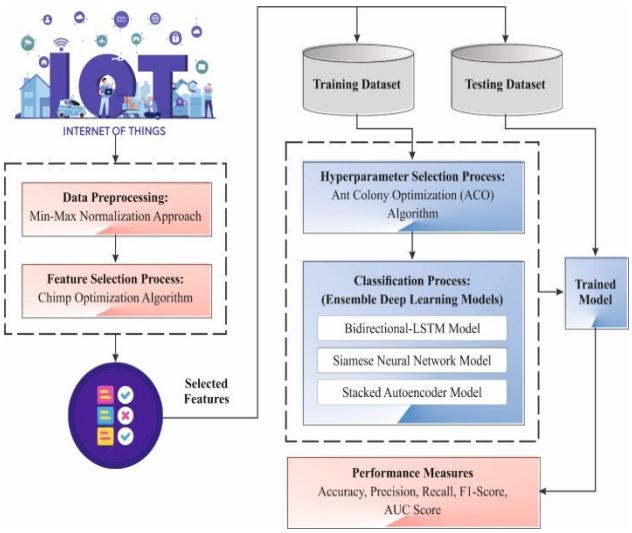

Figure 1 Overall workflow of OIDS-EDLMACO technique

3.2 ChOA-Based Feature Selection Process

Then, the FS process is implemented by the ChOA. The ChOA belongs to the swarm intelligence group meta-heuristics models, and it was advanced to follow the search model and cooperative behavioral patterns of a group of chimpanzees [19]. In this model, chimpanzees are separated into 4 main subcategories: chasers, attackers, callers, and holders all giving exclusively to improve the optimizer process. This collaborative model helps the method to keep a balance among exploitation (enhancing recent solutions) and exploration (look for new regions).

$$X_{\text{attack}} = X_{\text{best}} - A \cdot \left| C \cdot X_{\text{best}} - X \right| \tag{1}$$

Now, X_{best} signifies the best-performing chimp position, whereas A and C are similar to the coefficient vectors adaptively adjusted inside all rounds, authorizing the exploration process.

Individuals coming from the chasing group X_{chase} upgrade their locations as administered by Eq. (2):

$$X_{\text{chase}} = X_{\text{attack}} - B \cdot \left| D \cdot X_{\text{attack}} - X \right| \tag{2}$$

whereas B and D act as controller variables to maintain the balance between exploitation and exploration stages.

The individuals coming from the holder group X_{hold} refresh their locations in line by Eq. (3):

$$X_{\text{hold}} = X_{\text{chase}} - E \cdot \left| F \cdot X_{\text{chase}} - X \right| \tag{3}$$

Now, E and F perform as supplementary parameters, which manage this upgrade.

Lastly the individuals from caller group X_{call} implement location upgrade based on Eq. (4):

$$X_{\text{call}} = X_{\text{hold}} - G \cdot \left| H \cdot X_{\text{hold}} - X \right| \tag{4}$$

Now G and H have the same characters to A and C, attuned to the caller's function of the optimizer procedure.

By iteratively refining locations, ChOA utilizes the cooperative intelligence of the various chimpanzee characters to resolve composite optimizer tasks, prove become an effective model to deal with higher-dimensional searching areas inside a wide range of real-time applications. The fitness function (FF) applied in the ChOA model is intended to have a balance amongst the selected feature counts in all solutions (minimal) and the classification exactness (maximal) gained by utilizing these chosen characteristics, Eq. (5) embodies the FF to estimate solutions.

$$\text{Fitness} = \alpha \gamma_R(D) + \beta \frac{|R|}{|C|} \tag{5}$$

whereas $\gamma_R(D)$ measures the error rate of the classifier provided.

3.3 Ensemble-Based Classification Model

For the classification process, the proposed OIDS-EDLMACO model designs ensemble models such as BiLSTM method, SNN technique, and SAE model.

3.3.1 BiLSTM Classifier

LSTM is an RNN type applied to handle sequence data and time series problem. It was presented to rapidly obtain popularity afterward RNN resolved the problem of the gradient vanishing. LSTM was designed for addressing the problem that conventional RNNs bring in addressing long-range dependencies [20]. Once usual RNN handles longer sequences, the gradient gradually explodes or disappears, making longer-distance dependencies challenging to seizure. LSTM deals with this problem by adding a gating mechanism. The memory cell of the LSTM contains 3 portions: the input, the output, and the forget gates. C_{t-1} refers to cell state of the preceding moment, h_{t-1} stands for last output value of the LSTM component at the final instant, x_t represents input for the present instant, σ means activation function, f_1 symbolize output of the forget gate at the present instant, i_t signifies input gate output for the present instant, $\tilde{C}_t$ represents candidate cell status at the present instant, o_t denotes output gate's output value, C_t denotes cell state at the present instant, h_t indicates the present moment's output.

LSTM is a type of RNN used for dealing with sequence data and time series issues. Once the gradient vanishing problem was solved by RNN, LSTM was introduced to quickly gain popularity. LSTM was developed to solve the problem that RNNs have when dealing with long-range dependencies [20]. As usual RNNs handle longer sequences, the gradient either grows too large or becomes too small, making it tough to catch longer-distance connections. LSTM solves this issue by using a gating system. The memory cell in an LSTM consists of the input, the output and the forget gates.

3.3.2 SNN Classifier

The Siamese network is a unique architecture of deep neural networks (DNNs), containing two or several subnetworks, which concurrently get input feature data and sharing DNNs weights. Therefore, all subnetworks may

individually map the input into the latent feature area [21]. The basis of the Siamese NN's complete architecture is to discover a suitable mapping function $g(x)$. This mapping function can design the input procedure pairs of data ($x(t_1)$, $x(t_2)$) into the latent feature area($h(t_1)$, $h(t_2)$), guaranteeing that the simpler spaces (like cosine distance, Euclidean distance, and so on) among dissimilar instances in the latent feature area estimated the real natural relationships in the input area. In detail, the Siamese network architecture targets to upgrade parameters to discover a collection of expressions while the similarity measurement is lesser for two or several instances, which are nearly parallel, and greater for some other than similar. The Siamese network's objective function is defined as:

$$\text{loss} = \begin{cases} \left\| h(t_1) - h(t_2) \right\|_2^2, \text{ if } x(t_1) \text{ and } x(t_2) \text{ are similar} \\ \max\left(O, \delta - h(t_1) - h(t_2) \right\|_2^2 \right), \text{ otherwise.} \end{cases} \quad (6)$$

Here $\| h(t_1) - h(t_2) \|_2^2$ represents Euclidean distance metric, and the hyperparameter δ denotes minimal distance among the latent features of different instances. Consequently, the aim of the Siamese network is to reduce the distance amongst the latent features of dual instances. In network training, the dual network's parameters are upgraded concurrently in a related way, guaranteeing that the systems characterize the identical non-linear function.

3.3.3 SAE Classifier

SAE is the autoencoder (AE) configuration, which raises the sparsity limit for the loss function. At the same time, similar HL nodes are active, thus the entire AE system is intended sparsely, provide us utilize the activation function of the sigmoid HL [22]. The output of the HL utilizes 1 for the active node and 0 represents the inactive node. Based on this, the dispersion of K_L is recognized to calculate the similarities amongst the average activation outcome of a particular HL node and sparsity ρ as:

$$K_L(\rho \,||\, \hat{\rho}) = \rho \log \frac{\rho}{\hat{\rho}} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}} \quad (7)$$

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^{m} a_j(x_i) \quad (8)$$

whereas $\hat{\rho}_j$ indicates the average sparse activation, x_j and m represent the trained instances and sum of trained instances. $a_j(x)$ denotes the responding output of the j^{th} node of the HL to the i^{th} instance. Typically, the sparsity coefficient ρ is set to 0.1. The greater the divergence of KL, the greater the difference between ρ and $\hat{\rho}_j$, and the divergence of KL equivalent to 0 represents the two is completely equal. The dispersion of KL is the added secure term to utilize in the AE function for limiting the sparse rows of the entire AE:

$$\text{JSAC}(W, b) = J_{AE}(W, b) + \beta \sum_{j=1}^{m} K_L(\rho \,||\, \hat{\rho}_j) \quad (9)$$

whereas β suggests the sparse constraint's weighted coefficient. Fig. 2 represents the structure of SAE model.

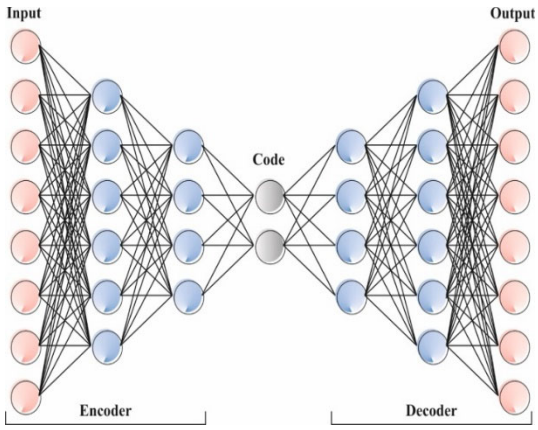

Figure 2 Structure of SAE model

3.4 ACO-Based Parameter Tuning Model

Additionally, the ACO algorithm-based hyperparameter selection process is performed to optimize the classification outcomes of ensemble classifiers. ACO is a nature inspired optimizer model that originated from the ant's hunting behavior naturally [23]. It mainly deals with combinative optimizer issues, like path planning and the Traveling Salesman Problem (TSP). This model outshines in determining optimum or near-optimum routes inside composite, multidimensional surroundings, making it appropriate for path planning in either dynamic or static situations. Its path selection mechanism, according to pheromone concentration and heuristic information, allows the fast recognition of the best solution from amongst numerous possible routes. Initially, parameter initialization has been carried out by setting the ant counts m, the pheromone's primary value τ_0, the pheromone significance feature α, the heuristic information significance feature β, the volatility coefficient of the pheromone ρ, and the maximal iteration counts T.

$$p_{ij}^{k} = \frac{\left[\tau_{ij}(t) \right]^{\alpha} \cdot \left[\eta_{ij} \right]^{b}}{\sum_{l \in \text{allowed\_nodes}} \left[\tau_{il}(t) \right]^{\alpha} \cdot \left[\eta_{il} \right]^{b}} \quad (10)$$

whereas $\tau_{ij}(t)$ denotes pheromone attentiveness on the route (i, j) and η_{ij} represents heuristic information, typically the path visibility. Pheromone upgrading contains pheromone volatilization and adding of novel pheromone.

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \Delta \tau_{ij} \quad (11)$$

whereas ρ denotes pheromone volatilization coefficient ($0 < \rho < 1$) that designates the pheromone decay in time. $\Delta\tau_{ij}$ denotes sum of recently added pheromone, controlled by the amount of pheromone left by each ant on the route.

$$\Delta \tau_{ij} = \sum_{k=1}^{m} \Delta \tau_{ij}^{k} \quad (12)$$

For the ant k, the sum of pheromone it goes on the route (i, j) is as demonstrated:

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k}, & \text{if } k \to (i,j) \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Here, Q denotes constant, and L_k means total route length of the solution made by ant, $kk \to (i, j)$ characterizes that ant k has passed over point (i, j) reiterating the above procedure of building the solution and upgrading the pheromone till a pre-determined maximal iteration counts T is attained or an acceptable solution finishes the optimum path planning.

4 EXPERIMENTAL ANALYSIS

The way the OIDS-EDLMACO approach was tested is analyzed in the dataset [24]. There are 25000 samples in the dataset and they are divided into two classes: normal and attack. At present, the database includes four types of attacks: aggressive scan, Sparta secure shell brute force, UDP scan and MQTT brute-force attack. All the information about this database is given in Tab. 1. There are 32 features in total, but only 21 features are chosen.

Table 1 Details of dataset

Class labels	Total Samples	Normal	Attack	For Experimental
Biflow_mqtt_bruteforce	1669	2152	14544	5000
Biflow_normal	86008	86008	X	5000
Biflow_scan_A	25693	5786	19907	5000
Biflow_scan_sU	39664	17230	22434	5000
Biflow_sparta	91318	77202	14116	5000
Total	259379	188378	71001	25000

Fig. 3 shows the confusion matrix that results from the OIDS-EDLMACO methodology with a TRASE/TESSE ratio of below 80:20% and 70:30%. The findings demonstrate that the OIDS-EDLMACO method works well in spotting and identifying all classes of malware.

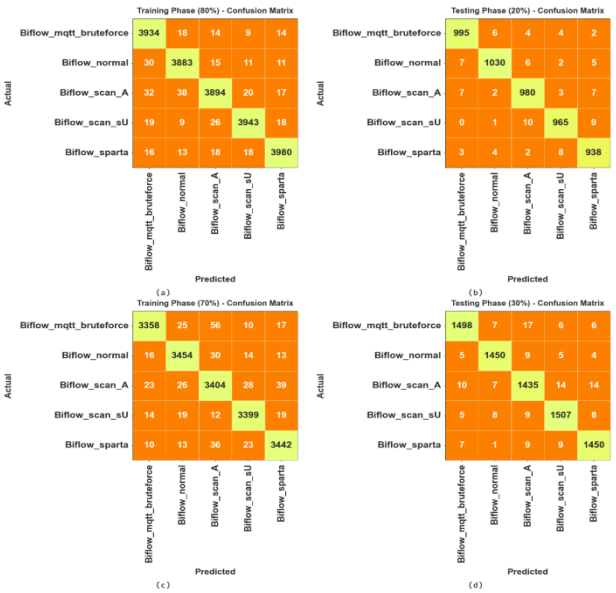

Figure 3 Confusion matrix of OIDS-EDLMACO algorithm (a-b) 80%TRASE and 20%TESSE and (c-d) 70%TRASE and 30%TESSE

Tab. 2 and Fig. 4 signify the intrusion detection of OIDS-EDLMACO approach under 80:20 and 70:30% of TRASE/TESSE.

Table 2 Intrusion detection of OIDS-EDLMACO model under 80:20 and 70:30 of TRASE/TESSE

Class labels	*ACY*	*PRE*	*REC*	*FS*	*AUC*
TRASE (80%)					
Biflow_mqtt_ bruteforce	99.24	97.59	98.62	98.10	99.01
Biflow_normal	99.28	98.03	98.30	98.17	98.91
Biflow_scan_A	99.10	98.16	97.33	97.74	98.43
Biflow_scan_sU	99.35	98.55	98.21	98.38	98.92
Biflow_sparta	99.38	98.51	98.39	98.45	99.01
Average	99.27	98.17	98.17	98.17	98.86
TESSE (20%)					
Biflow_mqtt_brutefo rce	99.34	98.32	98.42	98.37	99.00
Biflow_normal	99.34	98.75	98.10	98.42	98.88
Biflow_scan_A	99.18	97.80	98.10	97.95	98.77
Biflow_scan_sU	99.26	98.27	97.97	98.12	98.77
Biflow_sparta	99.20	97.61	98.22	97.91	98.83
Average	99.26	98.15	98.16	98.15	98.85
TRASE (70%)					
Biflow_mqtt_ bruteforce	99.02	98.16	96.88	97.52	98.22
Biflow_normal	99.11	97.65	97.93	97.79	98.67
Biflow_scan_A	98.57	96.21	96.70	96.46	97.87
Biflow_scan_sU	99.21	97.84	98.15	98.00	98.81
Biflow_sparta	99.03	97.51	97.67	97.59	98.52
Average	98.99	97.47	97.47	97.47	98.42
TESSE (30%)					
Biflow_mqtt_ bruteforce	99.16	98.23	97.65	97.94	98.60
Biflow_normal	99.39	98.44	98.44	98.44	99.03
Biflow_scan_A	98.81	97.03	96.96	96.99	98.11
Biflow_scan_sU	99.15	97.79	98.05	97.92	98.74
Biflow_sparta	99.23	97.84	98.24	98.04	98.85
Average	99.15	97.87	97.87	97.87	98.67

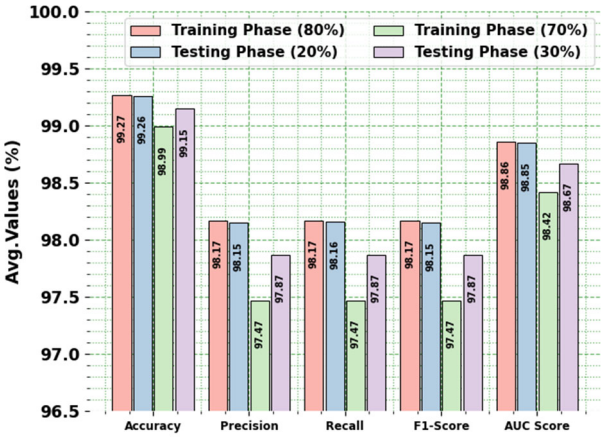

Figure 4 Average of OIDS-EDLMACO model under 80:20 and 70:30 of TRASE/TESSE

The analysis of the OIDS-EDLMACO model with different training-testing splits (80:20 and 70:30 for TRASE and TESSE) reveals that it can detect different types of attacks very well. The model shows an average accuracy of 99.27% on the training set and 99.26% on the test set and all the other metrics are close to 98.15-98.86%, suggesting that the model generalizes well and does not overfit much. Biflow_sparta and Biflow_normal achieved the best results, but Biflow_scan_A did not perform as well which shows that more work is needed on its features or the training data. Even with less data for training, the model

still performs well, reaching 98.99% and 99.15% accuracy on TRASE and TESSE and an average AUC of 98.42-98.67%, proving its stability. All the scores are still above 97.4% and no big change in performance is seen when more test data is used. All in all, the model is reliable in telling apart normal and malicious flows, keeping a good balance between false positives and false negatives, as shown by impressive F1 and AUC scores. The findings prove that OIDS-EDLMACO is a strong and usable framework for intrusion detection, but some improvements could be made to its recall for Biflow_scan_A attacks.

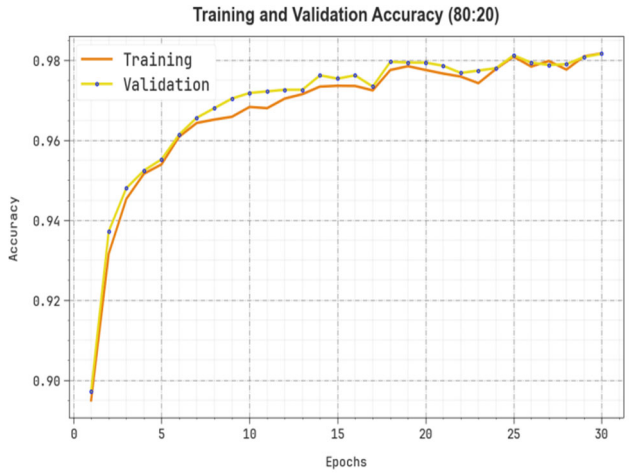

Figure 5 *Accu$_y$* curve of OIDS-EDLMACO method under 80% TRASE and 20% TESSE

Tab. 3 shows the outcomes of the OIDS-EDLMACO algorithm when compared with other methods using several metrics and training & testing times [3, 25-27].

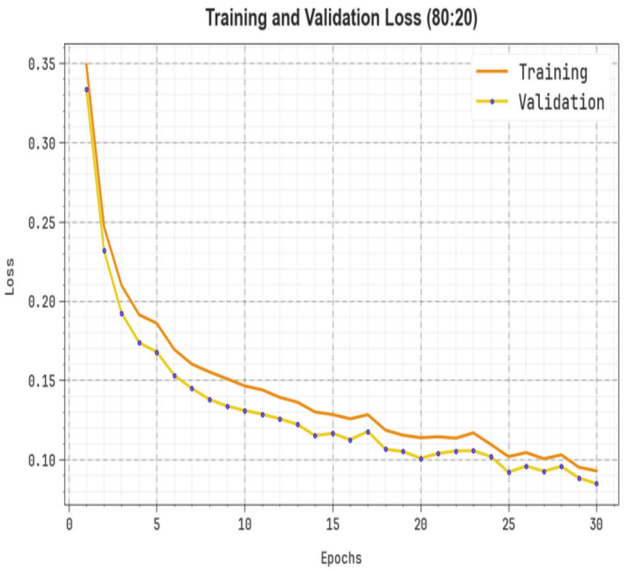

Figure 6 Loss analysis of OIDS-EDLMACO method below 80% TRASE and 20% TESSE

In Fig. 5, the OIDS-EDLMACO technique is shown to have training and validation accuracy below 80% and 20% respectively. The analysis is done for each of the 0-30 epochs. The graph proves that the TRAand VAL accu_y analysis shows a rise which allowed the OIDS-EDLMACO technique to perform well in different iterations. At the same time, the TRA and VAL results became more alike as the epochs passed which prevented overfitting and gave the best results from the OIDS-EDLMACO system, making sure hidden samples were predicted accurately.

Table 3 Comparative results of OIDS-EDLMACO technique with existing algorithms

Methodology	*Accuracy*	*Precision*	*Recal*	F_{score}	Training Time / sec	Testing Time / sec
K-NN	97.04	88.38	82.40	85.30	10.88	11.97
Naïve Bayes	91.25	92.07	79.43	85.28	6.72	6.88
Decision Tree	97.09	95.55	81.07	87.71	6.15	7.94
Random Forest	98.03	96.46	84.43	90.06	8.45	4.04
LSTM	93.44	97.05	92.57	93.12	8.04	4.51
GRUs	96.15	92.25	89.41	97.87	4.38	3.98
XGBoost	99.07	97.73	97.38	97.53	3.29	5.12
OIDS-EDLMACO	99.27	98.17	98.17	98.17	2.88	2.99

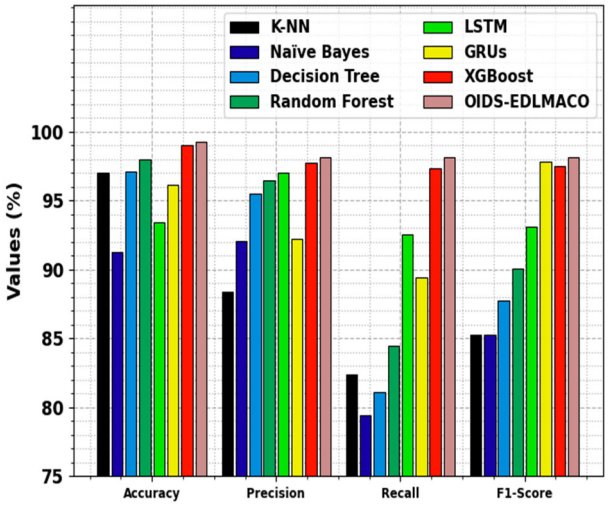

Figure 7 Comparative analysis of OIDS-EDLMACO model with existing algorithms

Fig. 7 compares the OIDS-EDLMACO methodology to other algorithms that have been used. According to the table, the OIDS-EDLMACO techniques have achieved better performance with prec_n, reca_l, accu_y and F_score of 98.17%, 98.17%, 99.27% and 98.17%, respectively. At the same time, the KNN, NB, DT, RF, LSTM, GRUs and XGBoost methods have shown lower performance.

Fig. 8 shows the training and testing times of OIDS-EDLMACO method compared to other models. The OIDS-EDLMACO system was found to have a training time of 2.88 sec and a testing time of 5.12 sec which is much less than that of KNN, NB, DT, RF, LSTM, GRUs and XGBoost.

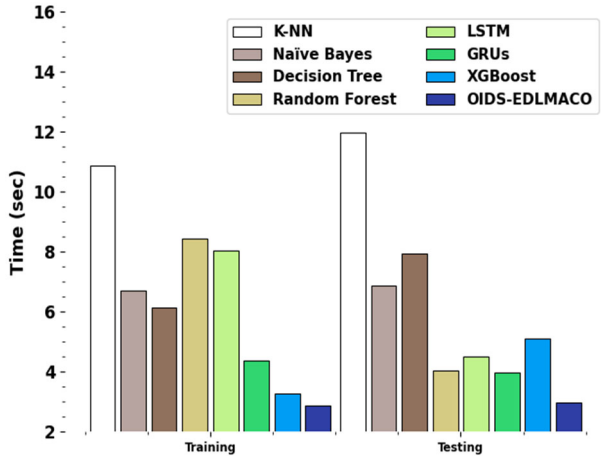

Figure 8 Training time and testing time of OIDS-EDLMACO model with existing algorithms

5 CONCLUSION

This paper suggests the use of an OIDS-EDLMACO technique. The goal of OIDS-EDLMACO is to enhance IoT network attack detection by using the latest ensemble models. At this stage, the data is pre-processed using min-max normalization to make it useful. After that, the ChOA carries out the FS process. Moreover, the OIDS-EDLMACO model designs ensembles like BiLSTM, SNN and SAE for the purpose of classification. The ACO algorithm is applied to help choose the best hyperparameters for improving how ensemble classifiers perform. The OIDS-EDLMACO algorithm can be tested through experiments on database. The many results prove that the OIDS-EDLMACO approach is very effective in intrusion detection.

6 REFERENCES

[1] Kavitha, S., Uma Maheswari, N., & Venkatesh, R. (2023). Intelligent Intrusion Detection System using Enhanced Arithmetic Optimization Algorithm with Deep Learning Model. *Technical Gazette*, *30*(4), 1217-1224. https://doi.org/10.17559/TV-20221128071759

[2] Rathee, G., Kerrache, C. A., & Ferrag, M. A. (2022). A blockchain-based intrusion detection system using viterbi algorithm and indirect trust for iiot systems. *Journal of Sensor and Actuator Networks*, *11*(4), 71. https://doi.org/10.3390/jsan11040071

[3] Wang, H., Wen, W., Zhang, Z., & Gao, N. (2023). Construction of Building Energy Consumption Prediction Model Based on Multi-Optimization Model. *Buildings*, *13*(7), 1677. https://doi.org/10.3390/buildings13071677

[4] Li, Y. & Zhanyong, W. (2023). A Cloud Based Network Intrusion Detection System. *Technical Gazette*, *29*(3), 987-992. https://doi.org/10.17559/TV-20211130024245

[5] Sharadqh, A. A., Hatamleh, H. A. M., Saloum, S. S., & Alawneh, T.A. (2023). Hybrid Chain: Blockchain Enabled Framework for Bi-Level Intrusion Detection and Graph-Based Mitigation for Security Provisioning in Edge Assisted IoT Environment. *IEEE Access*, *11*, 27433-27449. https://doi.org/10.1109/ACCESS.2023.3256277

[6] Alkadi, O., Moustafa, N., Turnbull, B., & Choo, K. K. R. (2020). A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks. *IEEE Internet of Things Journal*, *8*(12), 9463-9472. https://doi.org/10.1109/JIOT.2020.2996590

[7] Ashraf, E., Areed, N. F., Salem, H., Abdelhay, E. H., & Farouk, A. (2022), June. Fidchain: Federated intrusion detection system for blockchain-enabled iot healthcare applications. *Healthcare*, *10*(6), 1110. https://doi.org/10.3390/healthcare10061110

[8] Abdulqadder, I. H., Zou, D., & Aziz, I. T. (2023). The DAG blockchain: A secure edge assisted honeypot for attack detection and multi-controller based load balancing in SDN 5G. *Future Generation Computer Systems*, *141*(7), 339-354. https://doi.org/10.1016/j.future.2022.11.008

[9] N, K. & JS, S. M. (2023). BRDO: Blockchain Assisted Intrusion Detection Using Optimized Deep Stacked Network. *Cybernetics and Systems*, *55*(8), 2071-2092. https://doi.org/10.1080/01969722.2023.2175153

[10] Ahmed, M. A., Althubiti, S. A., Rao, D. N., Lydia, E. L., Cho, W., Joshi, G. P., & Kim, S. W. (2022). Blockchain Assisted Intrusion Detection System Using Differential Flower Pollination Model. *Computers, Materials & Continua*, *73*(3). 4695-4711. https://doi.org/10.32604/cmc.2022.032083

[11] Alevizos, L., Eiza, M. H., Ta, V. T., Shi, Q., & Read, J. (2022). Blockchain-enabled intrusion detection and prevention system of APTs within zero trust architecture. *IEEE Access*, *10*, 89270-89288. https://doi.org/10.1109/ACCESS.2022.3200165

[12] Derhab, A., Guerroumi, M., Gumaei, A., Maglaras, L., Ferrag, M. A., Mukherjee, M., & Khan, F. A. (2019). Blockchain and random subspace learning-based IDS for SDN-enabled industrial IoT security. *Sensors*, *19*(14), 3119. https://doi.org/10.3390/s19143119

[13] Liu, L. & Li, J. (2022). A Blockchain-assisted Collaborative Ensemble Learning for Network Intrusion Detection. *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 1042-1047. https://doi.org/10.1109/TrustCom56396.2022.00142

[14] Makhdoom, I., Zhou, I., Abolhasan, M., Lipman, J., & Ni, W. (2020). PrivySharing: A blockchain-based framework for privacy-preserving and secure data sharing in smart cities. *Computers & Security*, *88*, 101653. https://doi.org/10.1016/j.cose.2019.101653

[15] Guha Roy, D. & Srirama, S. N. (2021). A blockchain-based cyber attack detection scheme for decentralized Internet of Things using software-defined network. *Software: practice and experience*, *51*(7), 1540-1556. https://doi.org/10.1002/spe.2972

[16] Javeed, D., Gao, T., Saeed, M. S., & Kumar, P. (2023). An Intrusion Detection System for Edge-Envisioned Smart Agriculture in Extreme Environment. *IEEE Internet of Things Journal*, *11*(16), 26866-26876. https://doi.org/10.1109/JIOT.2023.3288544

[17] Meng, W., Li, W., Tug, S., & Tan, J. (2020). Towards blockchain-enabled single character frequency-based exclusive signature matching in IoT-assisted smart cities. *Journal of parallel and distributed computing*, *144*, 268-277. https://doi.org/10.1016/j.jpdc.2020.05.013

[18] Kumar, R., Kumar, P., Tripathi, R., Gupta, G. P., Garg, S., & Hassan, M. M. (2022). A distributed intrusion detection system to detect DDoS attacks in blockchain-enabled IoT network. *Journal of Parallel and Distributed Computing*, *164*, 55-68. https://doi.org/10.1016/j.jpdc.2022.01.030

[19] Abunadi, I., Althobaiti, M. M., Al-Wesabi, F. N., Hilal, A. M., Medani, M., Hamza, M. A., Rizwanullah, M., & Zamani, A. S. (2022). Federated learning with blockchain assisted image classification for clustered UAV networks. *Computer, materials & continua*, *72*(1), 1195-1212. https://doi.org/10.32604/cmc.2022.025473

[20] Vinoth Kumar, K. & Balaganesh, D. (2022). Efficient Privacy-Preserving Red Deer Optimization Algorithm with Blockchain Technology for Clustered VANET. *Technical Gazette*, *29*(3), 813-817. https://doi.org/10.17559/TV-20211216115635

[21] Liang, C., Shanmugam, B., Azam, S., Karim, A., Islam, A., Zamani, M., Kavianpour, S., & Idris, N. B. (2020). Intrusion detection system for the internet of things based on blockchain and multi-agent systems. *Electronics*, *9*(7), 1120. https://doi.org/10.3390/electronics9071120

[22] Zhang, X., Mo, T., & Zhang, Y. (2023). Optimization of Storage Location Assignment for Non-Traditional Layout Warehouses Based on the Firework Algorithm. *Sustainability*, *15*(13), 10242. https://doi.org/10.3390/su151310242

[23] Mafarja, M., Thaher, T., Al-Betar, M. A., Too, J., Awadallah, M. A., Abu Doush, I., & Turabieh, H. (2023). Classification framework for faulty-software using enhanced exploratory whale optimizer-based feature selection scheme and random forest ensemble learning. *Applied Intelligence*, *53*, 18715-18757. https://doi.org/10.1007/s10489-022-04427-x

[24] Balakrishnan, S. & Vinoth Kumar, K. (2023). Hybrid Sine-Cosine Black Widow Spider Optimization based Route Selection Protocol for Multihop Communication in IoT Assisted WSN. *Technical Gazette*, *30*(4), 1159-1165. https://doi.org/10.17559/TV-20230201000306.

[25] Wu, Q., Zhu, Q., & Han, S. (2023). Elman Neural Network-Based Direct Lift Automatic Carrier Landing Nonsingular Terminal Sliding Mode Fault-Tolerant Control System Design. *Computational Intelligence and Neuroscience*. https://doi.org/10.1155/2023/3560441

[26] Rahman, M. S., Khalil, I., Moustafa, N., Kalapaaking, A. P., & Bouras, A. (2021). A blockchain-enabled privacy-preserving verifiable query framework for securing cloud-assisted industrial internet of things systems. *IEEE Transactions on Industrial Informatics*, *18*(7), 5007-5017. https://doi.org/10.1109/TII.2021.3105527

[27] Cheema, M. A., Qureshi, H. K., Chrysostomou, C., & Lestas, M. (2020). Utilizing blockchain for distributed machine learning based intrusion detection in internet of things. *2020 16th international conference on distributed computing in sensor systems (DCOSS)*, 429-435. https://doi.org/10.1109/DCOSS49796.2020.00074

Contact information:

R. PANNEERSELVI
(Corresponding author)
Department of Computer Science and Engineering,
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology,
Chennai, India
E-mail: panneerselvirveltech@gmail.com

J. VISUMATHI, Professor
Department of Information Technology,
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology,
Chennai, India
E-mail: drvisumathij@veltech.edu.in