Computers, Materials & Continua
DOI: 10.32604/cmc.2023.032969
Article

Optimal Bottleneck-Driven Deep Belief Network Enabled Malware Classification on IoT-Cloud Environment

Mohammed Maray[1], Hamed Alqahtani[2], Saud S. Alotaibi[3], Fatma S. Alrayes[4], Nuha Alshuqayran[5], Mrim M. Alnfiai[6], Amal S. Mehanna[7] and Mesfer Al Duhayyim[8,*]

[1]Department of Information Systems, College of Computer Science, King Khalid University, Abha, Saudi Arabia
[2]Department of Information Systems, College of Computer Science, Center of Artificial Intelligence, Unit of Cybersecurity, King Khalid University, Abha, Saudi Arabia
[3]Department of Information Systems, College of Computing and Information System, Umm Al-Qura University, Saudi Arabia
[4]Department of information systems, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, P.O. Box 84428, Riyadh, 11671, Saudi Arabia
[5]Department of Information Systems, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University, Saudi Arabia
[6]Department of Information Technology, College of Computers and Information Technology, Taif University, P.O. Box 11099, Taif, 21944, Saudi Arabia
[7]Department of Digital Media, Faculty of Computers and Information Technology, Future University in Egypt, New Cairo, 11845, Egypt
[8]Department of Computer Science, College of Sciences and Humanities-Aflaj, Prince Sattam Bin Abdulaziz University, Saudi Arabia
*Corresponding Author: Mesfer Al Duhayyim. Email: m.alduhayyim@psau.edu.sa
Received: 02 June 2022; Accepted: 12 July 2022

Abstract: Cloud Computing (CC) is the most promising and advanced technology to store data and offer online services in an effective manner. When such fast evolving technologies are used in the protection of computer-based systems from cyberattacks, it brings several advantages compared to conventional data protection methods. Some of the computer-based systems that effectively protect the data include Cyber-Physical Systems (CPS), Internet of Things (IoT), mobile devices, desktop and laptop computer, and critical systems. Malicious software (malware) is nothing but a type of software that targets the computer-based systems so as to launch cyber-attacks and threaten the integrity, secrecy, and accessibility of the information. The current study focuses on design of Optimal Bottleneck driven Deep Belief Network-enabled Cybersecurity Malware Classification (OBDDBN-CMC) model. The presented OBDDBN-CMC model intends to recognize and classify the malware that exists in IoT-based cloud platform. To attain this, Z-score data normalization is utilized to scale the data into a uniform format. In addition, BDDBN model is also exploited for recognition and categorization of malware. To effectually fine-tune the hyperparameters related to BDDBN model, Grasshopper Optimization Algorithm (GOA) is applied. This scenario enhances the classification results and also shows the novelty of current study. The experimental analysis was conducted upon OBDDBN-CMC model for

validation and the results confirmed the enhanced performance of OBDDBN-CMC model over recent approaches.

Keywords: Malware detection; security; Internet of Things; cloud computing; machine learning; parameter adjustment

1 Introduction

Societies have become dependent upon technology in the past few years while technology is getting complicated day by day. In today's world, people and devices are heavily connected with each other. Especially, e-government, smart cities, smart homes and such data-driven technologies follow Internet of Things (IoT) model. After the outbreak of COVID-19 pandemic and the circumstances that led to continuous lockdowns, technology acceptance has augmented in multiple folds and through diverse methods. For example, e-health applications have been designed after COVID-19 outbreak to support the already- exhausted healthcare professionals and medical systems [1]. However, a comprehensive connection to the cyber-world, on the other hand, increased the amount of cyberattacks. These cyberattacks might reveal information which is generally categorized as confidential and secure. To be specific, Internet of Medical Things (IoMT) system deals with huge volumes of patient datasets and transmission and storage of medical data experienced severe privacy concerns [2]. As a result, certain benchmarks were developed in the meantime to overcome the shortcomings namely, implementation of secure data transmission protocol and ensuring the privacy of socket layers to avoid the leakage of private data [3]. Cybercrime is defined as any unauthorized activity that occurs upon computer or through conventional crime modes and target individuals or institutions via internet [4]. In this background, it has become inevitable to make one-stop security and trusted solution to handle information privacy and security in resource-constraint devices.

In general, IoT devices possess lesser processing and memory capacity which in turn makes the devices, lightweight [5]. These characteristics limit the predominant application of probable security solutions. When finding malware attacks in IoT environment, three predominant problems are faced [6]. Fig. 1 illustrates the security problems experienced in cloud-IoT. First of all, most of the IoT devices have low computation power which limits the complication of security system [7]. In addition, the intensification of hidden malware attacks that target the IoT systems, necessitates the quick adoption of detection method which in turn is a complicated approach [8,9]. Next, the rapid developments in IoT devices and the resultant security risks must be dealt with extremely strong data protection methods. Signature-based detection method plays a significant role in protecting the system from different types of malwares. This method gained much attention among researchers who conducted numerous studies focused upon its improvement in both academia and industries [10,11]. Signature is the concept built upon this concept of different malware detections. Signature is usually unalterable and identified in the earlier stages of propagation although the quantity of malware examples is constrained [12]. In this methodology, the content of the file is scanned and compared to check whether it matches the known signature [13].

Vasan et al. [14] suggested a new classification model for the detection of variants of malware groups and enhanced the detection of malware with the help of Convolutional Neural Network (CNN) based Deep Learning (DL) architecture named as Image-related Malware Classifier by making use of Finely tuned CNN structure (IMCFN). Being a novel technique, the solution varies from existing ones in terms of being a solution for multiclass classifier issues. Further, the technique converted the raw malware binary images into color images and the fine-tuned CNN structured utilized these images in

the detection and identification of malware groups. In the study conducted earlier [15], a combined DL technique was suggested for the detection of malware-infected documents and pirated software over IoT networks. Sudhakar et al. [16] devised a new malware classifier with fine-tuned (MCFT)-CNN method. The proposed MCFT-CNN method identified strange malware samples without any prior knowledge. In this study, reverse engineering method was followed with binary code analysis and even, the enhanced evading approaches were employed to detect the malwares.

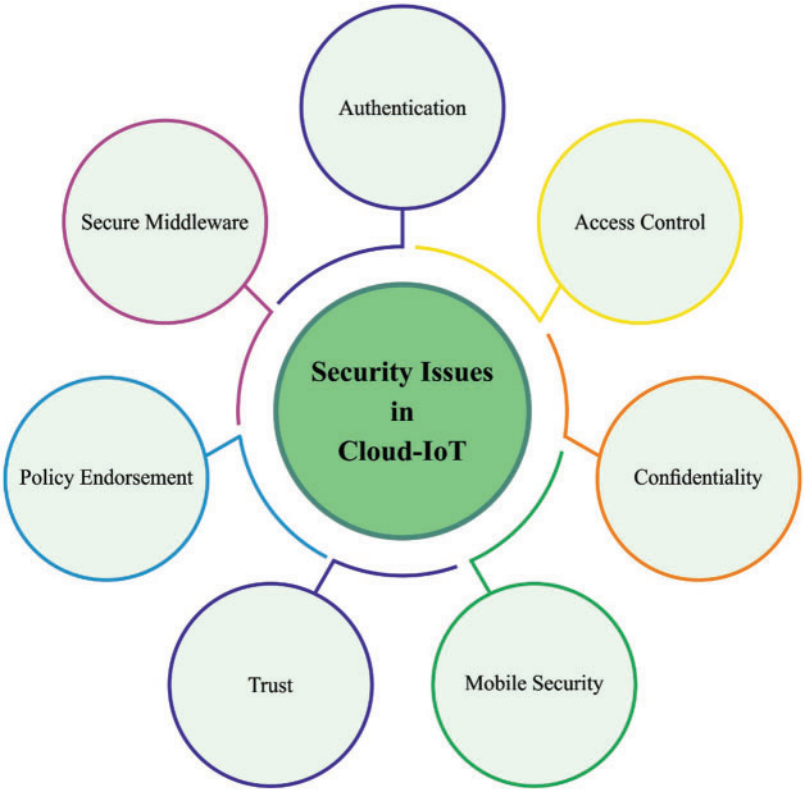

Figure 1: Security problems in cloud-IoT

Jeon et al. [17] recommended a Dynamic-Analysis-for-IoT-Malware-Detection (DAIMD) method to mitigate the damages caused in IoT gadgets through intelligent detection of familiar IoT malware and new variants of IoT malwares. DAIMD method investigated about IoT malwares with the help of CNN method and analyzed IoT malware vigorously in nested cloud atmosphere. DAIMD performed dynamic scrutinization of IoT malware in nested cloud setting so as to extract the behaviors based on virtual file system, memory, system call network, and process. In the study conducted earlier [18], a model was projected for detection of malware assaults on Industrial Internet of Things (MD-IIOT). For in-depth malware analysis, this study suggested a technique based on deep CNN color and image visualization. The outcomes achieved by the suggested technique were compared with that of the results from other studies in terms of malware detection.

The current study focuses on the design of Optimal Bottleneck driven Deep Belief Network-enabled Cybersecurity Malware Classification (OBDDBN-CMC) model. The aim of the presented OBDDBN-CMC model is to recognize and classify the malware in IoT-based cloud platform. To attain this, Z-score data normalization is utilized to scale the data into a uniform format. In addition,

BDDBN model is exploited for both recognition and categorization of the malware. To effectually fine-tune the hyperparameters related to BDDBN model, Grasshopper Optimization Algorithm (GOA) is used which in turn enhances the classification results. The proposed OBDDBN-CMC model was experimentally validated and the results confirmed the enhanced performance of OBDDBN-CMC model over recent approaches.

2 The Proposed OBDDBN-CMC Model

In current study, a novel OBDDBN-CMC model has been developed to recognize and classify the malware in IoT-based cloud platform. To attain this, Z-score data normalization is utilized to scale the data into a uniform format. In addition, BDDBN model is exploited for both recognition and categorization of malware. To effectually fine-tune the hyperparameters involved in BDDBN model, GOA is used which in turn enhances the classification results. Fig. 2 depicts the block diagram of OBDDBN-CMC approach.

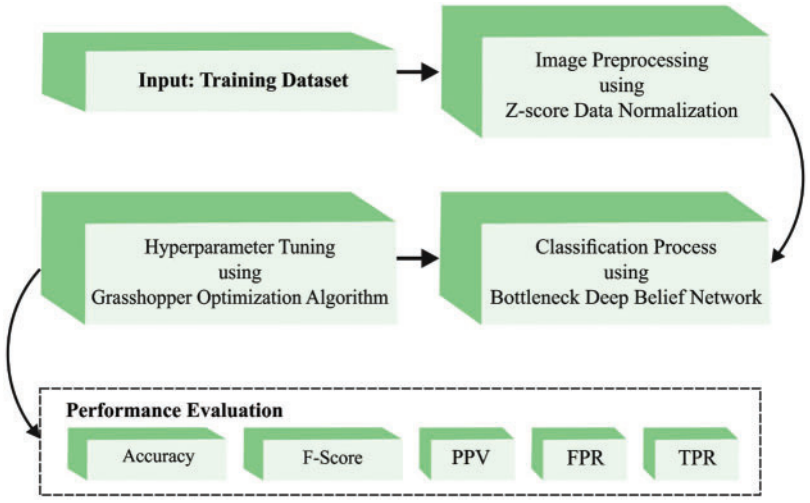

Figure 2: Block diagram of OBDDBN-CMC approach

2.1 Z-score Normalization

At first, Z-score data normalization is utilized to scale the data into a uniform format. *Z*-score is a traditional normalization and standardization technique that represents the number of Standard Deviations (SDs), in which · denotes a raw data point that is below or above population mean. It lies within the range of −3 and +3. Further, it standardizes the dataset based on the abovementioned scale in order to change every dataset with different scales to default scale.

In order to normalize the data using *z*-score, the mean of the population is subtracted from raw data point and divided using the SD. This yields a score that lies between −3 and +3, . thus, it reflects that SD is a point either below or above the mean as calculated using Eq. (1). In this equation, x refers to a certain sample's values, μ denotes the mean and σ indicates the SD [19].

$$z\_score = \frac{(x - \mu)}{\sigma} \tag{1}$$

2.2 BDDBN-Based Classification Process

After data pre-processing, BDDBN model is exploited for both recognition and categorization of malware [20]. DBN is a theoretical model applied in learning mechanism with deep structure. Deep structures denote that it contains multiple layers with non-linear arithmetical units. DBN has strong characterization and modelling ability and it can handle real-time datasets, for instance video, natural speech, and images, than the existing methods used for 'shallow' structure. Here, shallow structure denotes the individual layer with non-linear arithmetical units. Though DBN is basically a multi-layer Artificial Neural Network (ANN), it employs a hybrid of unsupervised and supervised training models to obtain the network parameters. This is to resolve the challenges faced by ANN-back propagation (BP) process in terms of getting trapped into local optima. Bottleneck concept is continuously employed for speech detection whereas BDDBN is the result of integrating bottleneck idea with DBN. BDDBN is generally established as a multi-layer ANN with odd number of layers whereas the middle layer is called 'bottleneck layer'. Bottleneck means the number of neurons in a layer is lower than other layers. BDDBN-based technique for speech feature extraction is executed as described below.

Step 1. Build DBN via fine-tuning and retraining and construct a nerve network.

Compositionally, DBN is a sequence of Restricted Boltzmann Machine (RBM) cascades. In general, RBM is composed of a hidden layer cell h_j and visible layer v_j connected whereas the joint distribution for a set of parameter models is formulated as follows.

$$E\left(v_i, h_j; \theta\right) = -\sum_{ij} w_{ij} v_j h_j - \sum_i b_i v_i - \sum_j a_j h_j, \tag{2}$$

$$E\left(v_j, h_j; \theta\right) = -\sum_{ij} w_{ij} v_j h_j - \sum_i b_j v_j - \sum_j a_j h_j,$$

Here, $\theta = \{w, a, b\}$ and w_{ij} refers to the connection weight of visible and hidden layers. b_j and a_j indicate the biases, correspondingly. The density likelihood distribution is defined by the equation given below.

$$P_\theta(v, h) = \frac{1}{Z(\theta)} \exp(-E(v, h; \theta)) \tag{3}$$

$$= \frac{1}{Z\left(\theta\right)} \prod_{ij} e^{w_{ij} v_i h_j} \prod_i e^{b_i v_j} \prod_j e_j^{ah},$$

where $Z(\theta) = \sum_{h,v} \exp\left(-E\left(v, h; \theta\right)\right)$, since the hidden node is conditionally independent of others as observed below.

$$P(h|v) = \prod_j P(h_j|v). \tag{4}$$

with the abovementioned formula, it is easy to obtain the probability of *j-th* node of the hidden unit i.e., one or zero as *v* visible layer.

$$P\left(h_j = 1|v\right) = \frac{1}{1 + \exp\left(-\sum_i w_{ij} v_i - a_j\right)},$$

$$P(v|h) = \prod_i P(v_j|h), \tag{5}$$

$$P\left(v_j = 1|h\right) = \frac{1}{1 + \exp\left(-\sum_j w_{ij} h_j - b_i\right)}.$$

Maximize the log-likelihood function:

$$L(\theta) = \frac{1}{N}\sum_{n=1}^{N} \log P_{\theta}(v) - \frac{\lambda}{N}\|w\|_F^2. \tag{6}$$

The derivation of maximal log likelihood function yields the w variable, equivalent to maximal L.

$$\frac{\partial L(\theta)}{\partial w_{ij}} = E_{P_{data}}[v_i h_j] - E_{P_{\theta}}[v_i h_j] - \frac{2\lambda}{N} w_{ij}. \tag{7}$$

A supervised learning mechanism is employed in current study compared to conventional Back Propagation Neural Network (BPNN) to build the whole DBN. In step 2, the bottleneck layer in the network is detached whereas the original bottleneck layer is applied as output layer.

2.3 GOA-Based Parameter Adjustment Process

To effectually fine-tune the hyperparameters related to BDDBN model, GOA is used which in turn enhances the classification results [21–23]. The fundamental basis of GOA is to mimic the behaviour of grasshopper during food search at adulthood and larval stages [24]. The behaviour of grasshopper swarms is statistically modelled as follows.

$$X_i = S_i + G_i + A_i \tag{8}$$

Let X_i be the location of *i-th* grasshopper, S_j indicates social interaction, G_j denotes the gravity force of *i-th* grasshopper, and A_i indicates the wind circulation.

$$S_j = \sum_{j=1, j\neq i}^{N} S\left(d_{i_j}\right) \overrightarrow{d_{i_j}} \tag{9}$$

In Eq. (9), d_{ij} refers to the distance between *i-th* and *j-th* grasshoppers, computed by $d_{i_j} = |X_j - X_i|$, S is a function utilized for defining robustness as given below, and $\overrightarrow{d_{i_j}} = \frac{X_j - X_i}{d_{ij}}$ indicates a unit vector between the function and *j-th* grasshopper.

The function S is calculated with the help of Eq. (10):

$$s(r) = fe^{\frac{-r}{l}} - e^{-r} \tag{10}$$

In this expression, f indicates attraction intensity and l denotes attraction scale length. The two parameters split the space function between two grasshoppers into regions of repulsion, comfort, and attraction. This S function relates to the robust force among grasshoppers, once the distance amongst themselves is lower.

The distance amongst the grasshoppers is standardized in the range of [1,4]. The G components of (8) are estimated as follows.

$$G_i = -g\vec{e}_g \tag{11}$$

where g represents the gravitational constant and $\vec{e}_g$ designates the unit vector toward the center of earth.

A component of (8) is evaluated as given below.

$$A_i = u\vec{e}_w \tag{12}$$

In Eq. (12), u denotes the adrift constant and $\vec{e}_w$ indicates the unit primarily in wind direction. The movement of the Nymph grasshopper primarily relies on wind's direction. Substitute G, and A in (13) and the resultant formula is given below.

$$X_i = \sum_{j=1, j\neq i}^{N} S(|X_j - X_i|)\frac{X_j - X_i}{d_{i_j}} - g\vec{e}_g + u\vec{e}_w \tag{13}$$

If $s(r) = fe^{\frac{-r}{l}} - e^{-r}$, then N indicates the overall number of grasshoppers. While the searching agent directs the sea toward a potential point, the quick reach of the grasshopper to comfort region is constrained as follows.

$$X_i^d = c \times \left(\sum_{j=1, j\neq 1}^{N} c \times \frac{ub_d - lb_d}{2} s\left(|X_j^d - X_i^d|\right) \frac{X_j - X_I}{d_{i_j}} 1 + \vec{I}_d \right) \tag{14}$$

Algorithm 1: Pseudo-code of GOA

Parameter initialization: population size (N), c_{max}, c_{min} and maximal amount of iteration ($t_{\max}$)
Generate population (X) arbitrarily
Fix curretn iteration $t = 1$
while ($t < t_{\max}$) do
    Determine fitness function f
    Select optimum solution $\hat{T}_d$
    Update value of (c)
    for $i = 1 : N$ do
        Normalize the distance among the solutions in X.
        Upgrade $x_i \in X$ by Eq. (12)
    end for
        $t = t + 1$
end while
Return $\hat{T}_d$.
where t

The initial term of this formula contemplates the position of existing grasshopper with regards to another grasshopper. Next, $\vec{I}_d$ simulates the tendency to move towards the food sources. ub_d and lb_d indicate the upper and lower limits of *d-th* parameter, $s(r) = fe^{\frac{-r}{l}} - e^{-r}$ is compared to (and I_d denotes the target value of *d-th* dimension, and c denotes the decreased coefficient which is employed for contracting the attraction, comfort, and repulsion regions. But, it is assumed that the direction of wind is often in the direction of target ($\vec{I}_d$). Both inner and outer c in (14) are similar to inertia weight factor w in Particle Swarm Optimization (PSO) as follows.

$$c = c_{\max} - iter\frac{c_{max} - c_{min}}{maxiter} \tag{15}$$

In Eq. (15), $c_{\max}$ represents the maximal value, $c_{\min}$ denotes the minimal value, *iter* and *maxiter* indicate the existing iteration and the maximal number of iterations, correspondingly. The outer c weakens the attraction and repulsion forces among the grasshoppers which is proportionate to the iteration count. The factor $c \times \frac{ub_d - lb_d}{2}$ linearly shrinks the space that needs to be exploited and explored by the grasshoppers. The factor $s(|X_j^d - X_i^d|)$ hints that a grasshopper must be repelled or attracted toward the exploitation phase.

GOA system resolves a Fitness Function (FF) to achieve maximum classification efficiency. It resolves a positive integer to portray the best efficiency of candidate results. In this case, minimization classifier error rate, supposedly to be provided by FF is given in Eq. (16).

$$fitness\,(x_i) = ClassifierErrorRate\,(x_i)$$

$$= \frac{number\ of\ misclassified\ samples}{Total\ number\ of\ samples} * 100 \tag{16}$$

3 Performance Validation

The presented OBDDBN-CMC model was experimentally validated for its performance using a dataset that contains 9,419 samples under two classes as demonstrated in Tab. 1.

Table 1: Dataset details

Class name	No. of samples
Benign	5065
Malware	4354
Total	9419

Fig. 3 illustrates the confusion matrices generated by OBDDBN-CMC model on the applied dataset with distinct training (TR) and testing (TS) data. With 70% of TR data, the proposed OBDDBN-CMC model recognized 3,455 samples as benign class and 2926 samples as malware class. Eventually, with 30% of TS data, the proposed OBDDBN-CMC approach categorized 1,476 samples under benign class and 1,276 samples under malware class. Meanwhile, with 80% of TR data, OBDDBN-CMC system classified 4,032 samples under benign class and 3,428 samples under malware class. At last, with 20% of TS data, the proposed OBDDBN-CMC methodology recognized 989 samples as benign class and 877 samples under malware class.

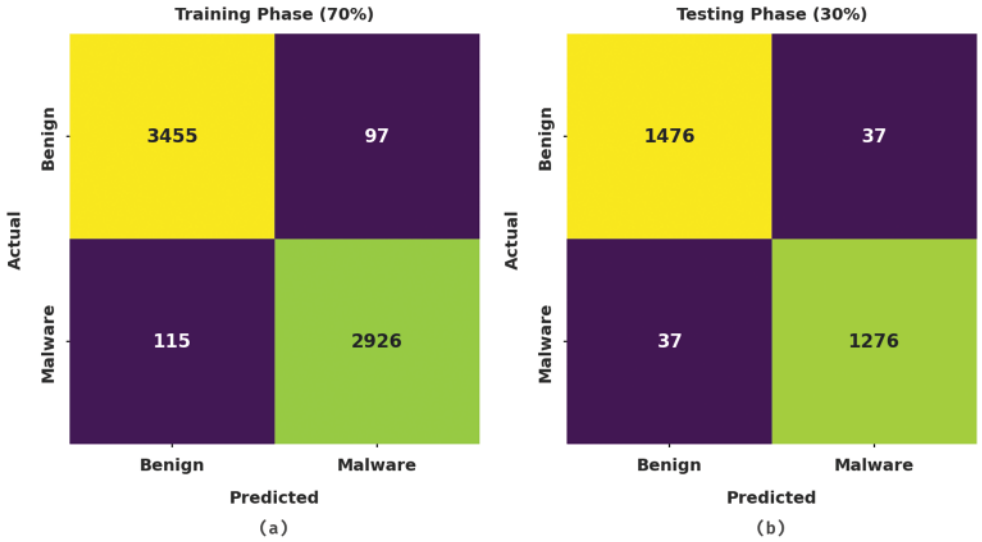

Figure 3: (Continued)

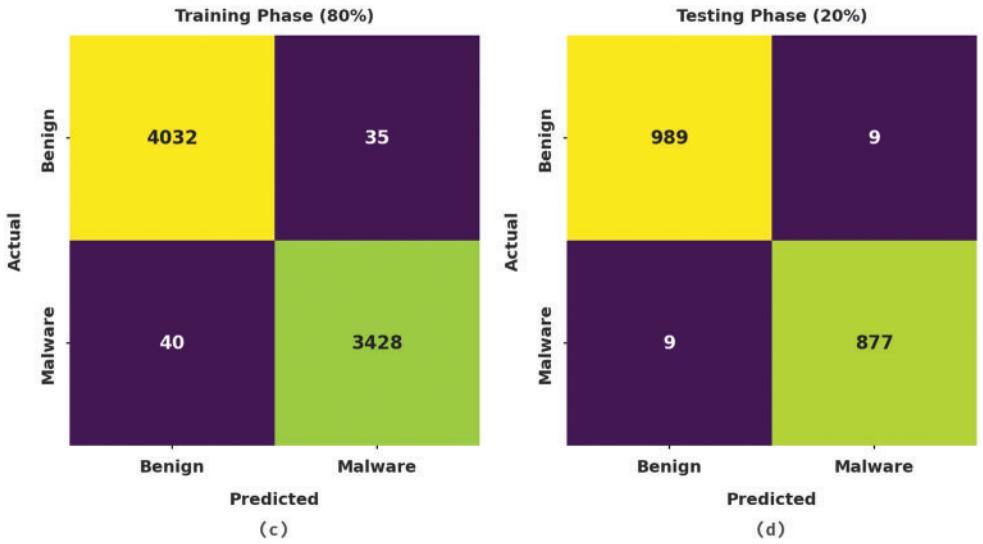

Figure 3: Confusion matrices of OBDDBN-CMC approach (a) 70% of TR data, (b) 30% of TS data, (c) 80% of TR data, and (d) 20% of TS data

Tab. 2 and Fig. 4 highlight the overall malware classification performance achieved by the proposed OBDDBN-CMC model under distinct aspects. With 70% of TR data, the proposed OBDDBN-CMC model offered an average $accu_y$ of 96.78%, Positive Predictive Value (PPV) of 96.78%, True Positive Rate (TPR) of 96.74%, F_{score} of 96.76%, and False Positive Rate (FPR) of 3.26%. Along with that, with 30% of TS data, the proposed OBDDBN-CMC system obtained an average $accu_y$ of 97.38%, PPV of 97.37%, TPR of 97.37%, F_{score} of 97.37%, and FPR of 2.63%. Moreover, with 80% of TR data, OBDDBN-CMC technique achieved an average $accu_y$ of 99%, PPV of 99%, TPR of 98.99%, F_{score} of 99%, and FPR of 1.01%. At last, with 20% of TS data, OBDDBN-CMC approach obtained an average $accu_y$ of 99.04%, PPV of 99.04%, TPR of 99.04%, F_{score} of 99.04%, and FPR of 0.96%.

Table 2: Results of the analysis of OBDDBN-CMC approach under different measures

Class labels	Accuracy	PPV	TPR	F-score	FPR
Training phase (70%)					
Benign	96.78	96.78	97.27	97.02	03.78
Malware	96.78	96.79	96.22	96.50	02.73
Average	96.78	96.78	96.74	96.76	03.26
Testing phase (30%)					
Benign	97.38	97.55	97.55	97.55	02.82
Malware	97.38	97.18	97.18	97.18	02.45
Average	97.38	97.37	97.37	97.37	02.63
Training phase (80%)					
Benign	99.00	99.02	99.14	99.08	01.15

(Continued)

Table 2: Continued

Class labels	Accuracy	PPV	TPR	F-score	FPR
Malware	99.00	98.99	98.85	98.92	00.86
Average	99.00	99.00	98.99	99.00	01.01
Testing phase (20%)					
Benign	99.04	99.10	99.10	99.10	01.02
Malware	99.04	98.98	98.98	98.98	00.90
Average	99.04	99.04	99.04	99.04	00.96

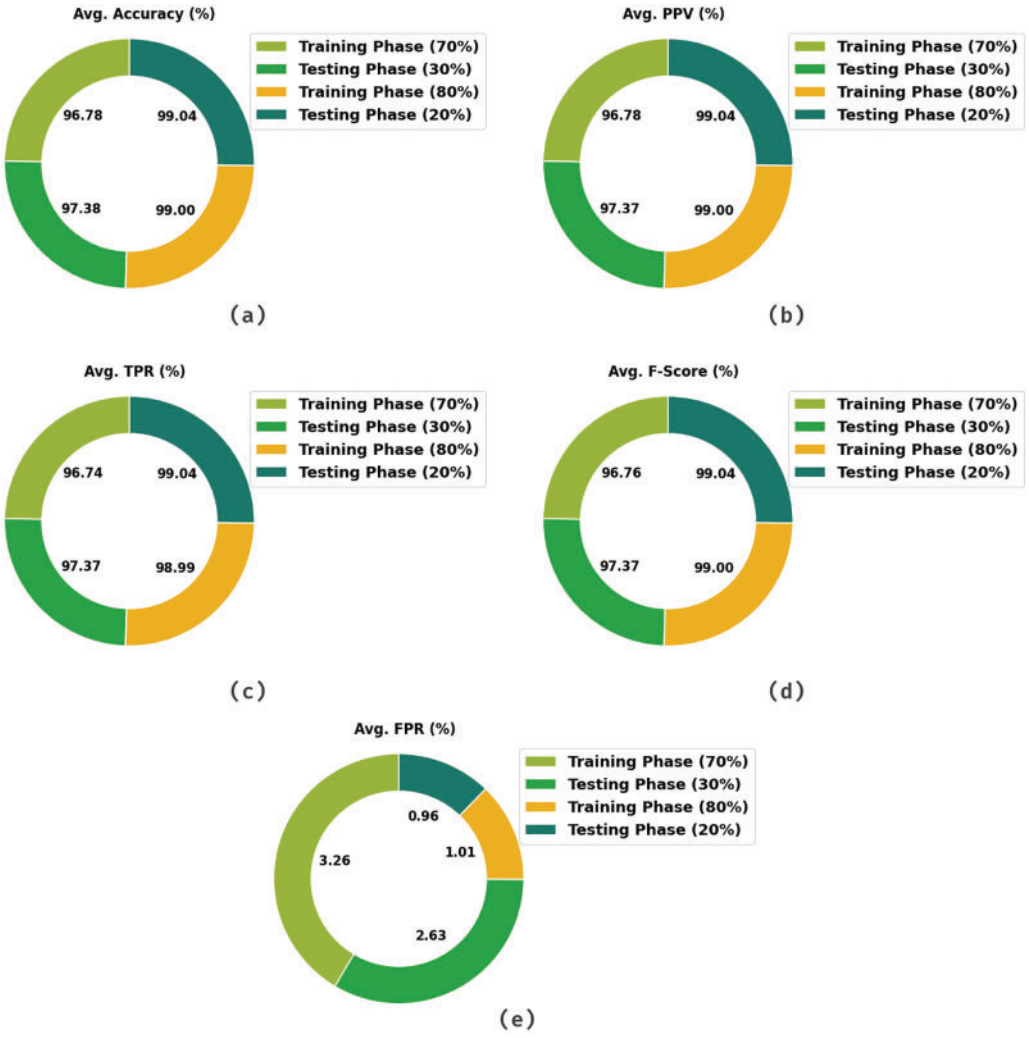

Figure 4: Average analysis results of OBDDBN-CMC approach (a) $Accu_y$, (b) PPV, (c) TPR, (d) F_{score}, and (e) FPR

A clear precision-recall inspection was conducted for OBDDBN-CMC method upon test dataset and the results are shown in Fig. 5. The figure implies that the proposed OBDDBN-CMC method produced enhanced precision-recall values under all the classes.

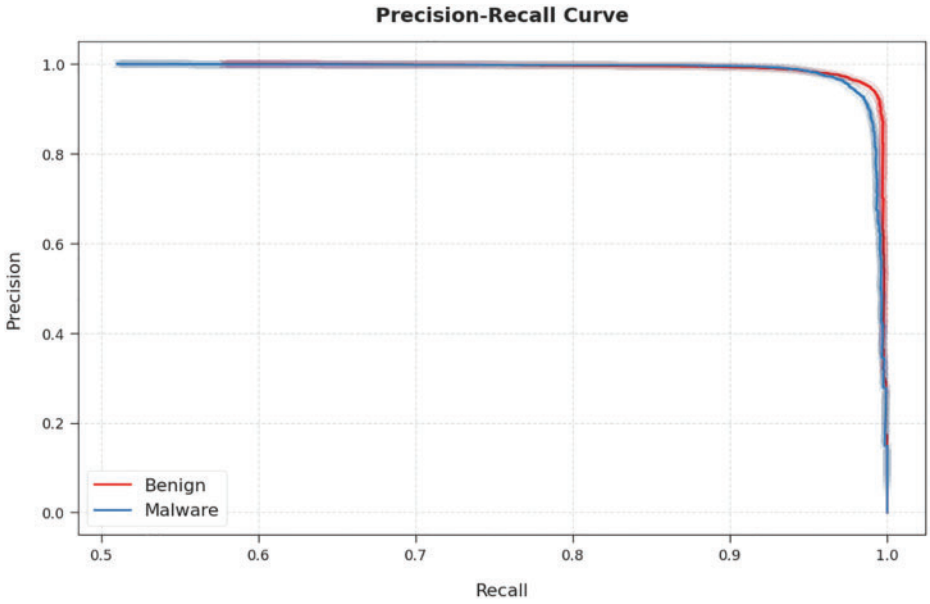

Figure 5: Precision-recall curve analysis of OBDDBN-CMC algorithm

A brief Receiver Operating Characteristic (ROC) curve analysis was conducted for OBDDBN-CMC method on test dataset and the results are depicted in Fig. 6. The results represent that the proposed OBDDBN-CMC approach revealed its ability to categorize distinct classes on test dataset.

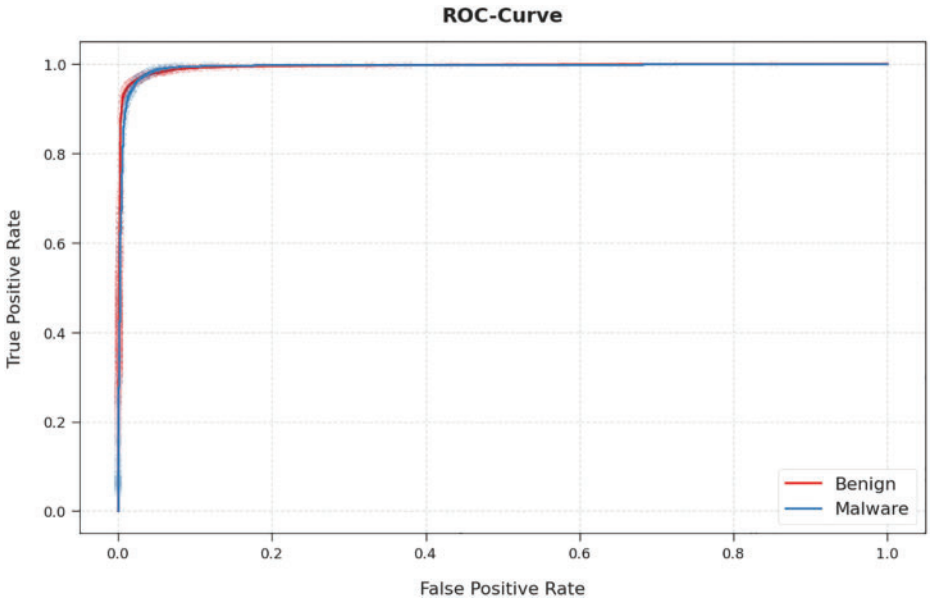

Figure 6: ROC curve of OBDDBN-CMC algorithm

Tab. 3 illustrates the comparative malware classification performance accomplished by the proposed OBDDBN-CMC model and other existing models [25,26] such as Improved Naïve Bayes (SVM), K-Nearest Neighbor (KNN), Naive Bayes (NB), DBN, TDS-GBAMD, EAMD-NF, and Two-Layer DL-android Malware Detection using Network Traffic (AMDNT).

Table 3: Comparative analysis results of OBDDBN-CMC approach and other existing methodologies

Methods	TPR	FPR	Accuracy	F-score
OBDDBN-CMC	99.04	0.96	99.04	99.04
Improved Naïve Bayes	98.21	1.67	97.81	97.64
SVM	95.02	5.03	95.18	97.96
KNN	75.75	12.68	92.00	90.83
Naive Bayes	80.25	18.98	90.36	88.78
DBN	85.00	14.70	86.69	86.56
TDS-GBAMD	81.27	18.82	84.19	83.37
EAMD-NF	87.59	11.90	87.46	88.48
Two-Layer DL-AMDNT	93.68	7.35	93.98	94.66

Fig. 7 highlights the analytical result accomplished by the proposed OBDDBN-CMC model and other existing models in terms of TPR, $accu_y$, and F_{score}. The results infer that the proposed OBDDBN-CMC model produced enhanced classification results. In terms of TPR, the proposed OBDDBN-CMC model offered an increased TPR of 99.04%, whereas INB, SVM, KNN, NB, DBN, TDS-GBAMD, EAMD-NF, and Two-layer DL-AMDNT models attained the least TPR values such as 98.21%, 95.02%, 75.75%, 80.25%, 85.00%, 81.27%, 87.59%, and 93.68% respectively. In terms of $accu_y$, the presented OBDDBN-CMC approach obtained the highest $accu_y$ of 99.04%, whereas INB, SVM, KNN, NB, DBN, TDS-GBAMD, EAMD-NF, and Two-layer DL-AMDNT approaches achieved the least $accu_y$ values such as 97.81%, 95.18%, 92%, 90.36%, 86.69%, 84.19%, 87.46%, and 93.98% correspondingly. At last, based on F_{score}, OBDDBN-CMC system obtained an increased F_{score} of 99.04%, whereas INB, SVM, KNN, NB, DBN, TDS-GBAMD, EAMD-NF, and Two-layer DL-AMDNT methodologies attained the least F_{score} values such as 97.64%, 97.96%, 90.83%, 88.78%, 86.56%, 83.37%, 88.48%, and 94.66% correspondingly.

A detailed FPR examination was conducted between the proposed OBDDBN-CMC model and other existing models in terms of FPR and the results are shown in Fig. 8. The figure implies that the proposed OBDDBN-CMC model gained low FPR values compared to other models. To be specific, OBDDBN-CMC model reached a low FPR of 0.96%, whereas INB, SVM, KNN, NB, DBN, TDS-GBAMD, EAMD-NF, and Two-layer DL-AMDNT models attained high FPR values such as 1.67%, 5.03%, 12.68%, 18.98%, 14.70%, 18.82%, 11.90%, and 7.35% respectively. Thus, the proposed OBDDBN-CMC model is found to have effectual malware classification efficiency in IoT-enabled cloud environment.

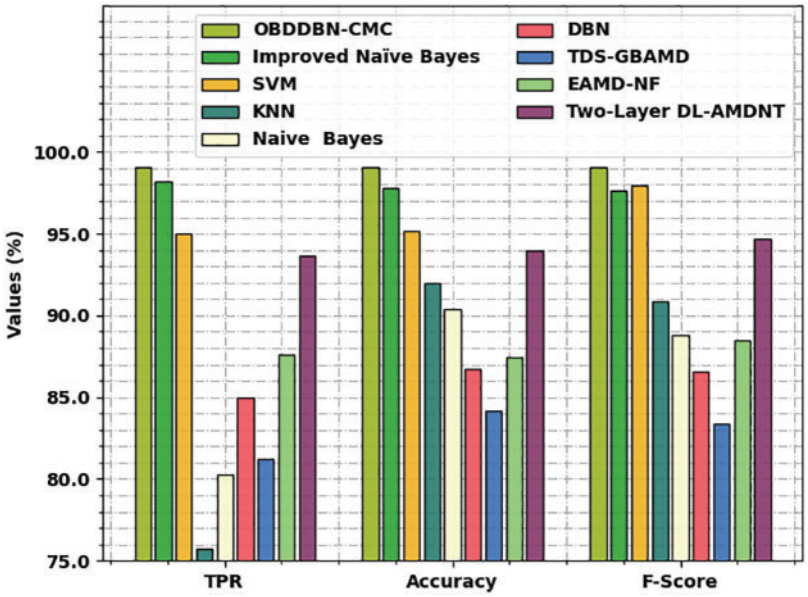

Figure 7: Comparative analysis results of OBDDBN-CMC approach and other existing methodologies

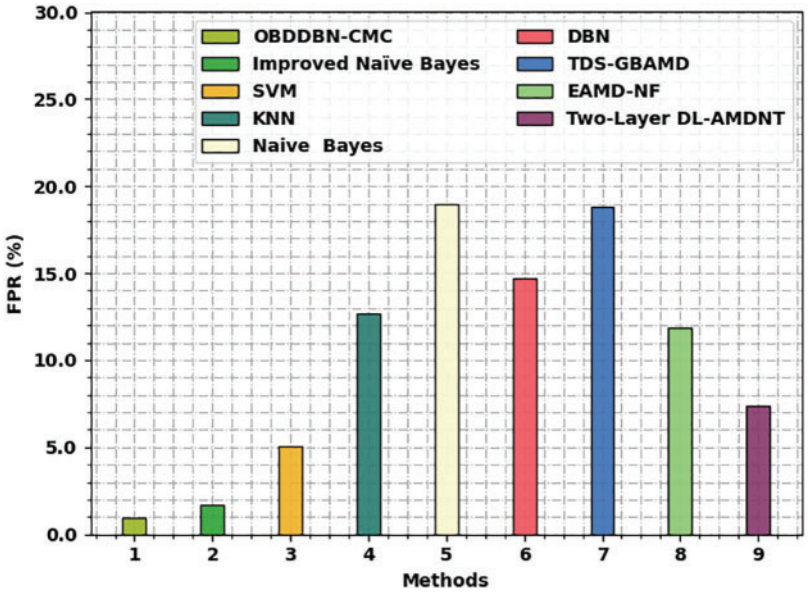

Figure 8: FPR analysis results of OBDDBN-CMC approach and other existing methodologies

4 Conclusion

In current study, a novel OBDDBN-CMC model has been developed to recognize and classify the malware in IoT-based cloud platform. To attain this, Z-score data normalization is utilized to scale the data into a uniform format. In addition, BDDBN model is exploited for recognition and categorization of malware. To effectually fine-tune the hyperparameters involved in BDDBN model, GOA is used which in turn enhances the classification results. The proposed OBDDBN-CMC model was experimentally validated and the results confirmed the enhanced performance of OBDDBN-CMC model over recent approaches. Thus, the presented OBDDBN-CMC model can be exploited as an effectual model for malware classification. In future, feature selection approaches can be designed to improve the classification performance.

Funding Statement: The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through Large Groups Project under grant number (61/43). Princess Nourah Bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R319), Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia. The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: (22UQU4210118DSR24).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Ö. Aslan, M. O. Okay and D. Gupta, "A review of cloud-based malware detection system: Opportunities, advances and challenges," *European Journal of Engineering and Technology Research*, vol. 6, no. 3, pp. 1–8, 2021.

[2] S. Zhao, S. Li, L. Qi and L. D. Xu, "Computational intelligence enabled cybersecurity for the Internet of Things," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 5, pp. 666–674, 2020.

[3] E. M. Dovom, A. Azmoodeh, A. Dehghantanha, D. E. Newton, R. M. Parizi *et al.,* "Fuzzy pattern tree for edge malware detection and categorization in IoT," *Journal of Systems Architecture*, vol. 97, no. 7, pp. 1–7, 2019.

[4] J. C. S. Sicato, P. K. Sharma, V. Loia and J. H. Park, "VPNFilter malware analysis on cyber threat in smart home network," *Applied Sciences*, vol. 9, no. 13, pp. 2763, 2019.

[5] Y. Shah and S. Sengupta, "A survey on classification of Cyber-attacks on IoT and IIoT devices," in *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conf. (UEMCON)*, New York, NY, USA, pp. 0406–0413, 2020.

[6] A. Al-Qarafi, F. Alrowais, S. Alotaibi, N. Nemri, F. N. Al-Wesabi *et al.,* "Optimal machine learning based privacy preserving blockchain assisted Internet of Things with smart cities environment," *Applied Sciences*, vol. 12, no. 12, pp. 1–17, 2022.

[7] M. Ficco, "Detecting IoT malware by Markov chain behavioral models," in *2019 IEEE Int. Conf. on Cloud Engineering (IC2E)*, Prague, Czech Republic, pp. 229–234, 2019.

[8] A. A. Albraikan, S. B. H. Hassine, S. M. Fati, F. N. Al-Wesabi, A. M. Hilal *et al.,* "Optimal deep learning-based cyberattack detection and classification technique on social networks," *Computers, Materials & Continua*, vol. 72, no. 1, pp. 907–923, 2022.

[9] M. Chikapa and A. P. Namanya, "Towards a fast off-line static malware analysis framework," in *2018 6th Int. Conf. on Future Internet of Things and Cloud Workshops (FiCloudW)*, Barcelona, pp. 182–187, 2018.

[10] I. Abunadi, M. M. Althobaiti, F. N. Al-Wesabi, A. M. Hilal, M. Medani *et al.,* "Federated learning with blockchain assisted image classification for clustered UAV networks," *Computers, Materials & Continua*, vol. 72, no. 1, pp. 1195–1212, 2022.
[11] U. Inayat, M. F. Zia, S. Mahmood, H. M. Khalid and M. Benbouzid, "Learning-based methods for cyberattacks detection in IoT systems: A survey on methods, analysis, and future prospects," *Electronics*, vol. 11, no. 9, pp. 1502, 2022.
[12] C. C. Uchenna, N. Jamil, R. Ismail, L. K. Yan and M. A. Mohamed, "Malware threat analysis techniques and approaches for IoT applications: A review," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 3, pp. 1158–1571, 2021.
[13] P. Ahirao, "Proactive technique for securing smart cities against malware attacks using static and dynamic analysis," *International Research Journal of Innovations in Engineering and Technology*, vol. 5, no. 2, pp. 10, 2021.
[14] D. Vasan, M. Alazab, S. Wassan, H. Naeem, B. Safaei *et al.,* "IMCFN: Image-based malware classification using fine-tuned convolutional neural network architecture," *Computer Networks*, vol. 171, no. 1, pp. 107138, 2020.
[15] F. Ullah, H. Naeem, S. Jabbar, S. Khalid, M. A. Latif *et al.,* "Cyber security threats detection in Internet of Things using deep learning approach," *IEEE Access*, vol. 7, pp. 124379–124389, 2019.
[16] Sudhakar and S. Kumar, "MCFT-CNN: Malware classification with fine-tune convolution neural networks using traditional and transfer learning in Internet of Things," *Future Generation Computer Systems*, vol. 125, no. 5, pp. 334–351, 2021.
[17] J. Jeon, J. H. Park and Y. S. Jeong, "Dynamic analysis for IoT malware detection with convolution neural network model," *IEEE Access*, vol. 8, pp. 96899–96911, 2020.
[18] H. Naeem, F. Ullah, M. R. Naeem, S. Khalid, D. Vasan *et al.,* "Malware detection in industrial Internet of Things based on hybrid image visualization and deep learning model," *Ad Hoc Networks*, vol. 105, no. 1, pp. 102154, 2020.
[19] E. Walia and A. Pal, "Fusion framework for effective color image retrieval," *Journal of Visual Communication and Image Representation*, vol. 25, no. 6, pp. 1335–1348, 2014.
[20] G. H. de Rosa and J. P. Papa, "Soft-tempering deep belief networks parameters through genetic programming," *Journal of Artificial Intelligence and Systems*, vol. 1, no. 1, pp. 43–59, 2019.
[21] I. V. Pustokhina, D. A. Pustokhin, E. L. Lydia, P. Garg, A. Kadian *et al.,* "Hyperparameter search based convolution neural network with Bi-LSTM model for intrusion detection system in multimedia big data environment," *Multimedia Tools and Applications*, vol. 13, no. 5, pp. 111, 2021. https://doi.org/10.1007/s11042-021-11271-7.
[22] G. N. Nguyen, N. H. L. Viet, M. Elhoseny, K. Shankar, B. B. Gupta *et al.,* "Secure blockchain enabled Cyber-physical systems in healthcare using deep belief network with ResNet model," *Journal of Parallel and Distributed Computing*, vol. 153, no. 2, pp. 150–160, 2021.
[23] M. N. A. Mhiqani, R. Ahmad, Z. Z. Abidin, K. H. Abdulkareem, M. A. Mohammed *et al.,* "A new intelligent multilayer framework for insider threat detection," *Computers & Electrical Engineering*, vol. 97, no. 1, pp. 107597, 2022.
[24] S. Dwivedi, M. Vardhan and S. Tripathi, "Building an efficient intrusion detection system using grasshopper optimization algorithm for anomaly detection," *Cluster Computing*, vol. 24, no. 3, pp. 1881–1900, 2021.
[25] R. Kumar, X. Zhang, W. Wang, R. U. Khan, J. Kumar *et al.,* "A multimodal malware detection technique for android IoT devices using various features," *IEEE Access*, vol. 7, pp. 64411–64430, 2019.
[26] J. Feng, L. Shen, Z. Chen, Y. Wang and H. Li, "A two-layer deep learning method for android malware detection using network traffic," *IEEE Access*, vol. 8, pp. 125786–125796, 2020.