Mobile Networks and Applications (2023) 28:795–807
https://doi.org/10.1007/s11036-023-02105-x

Quantum Mayfly Optimization with Encoder-Decoder Driven LSTM Networks for Malware Detection and Classification Model

Omar A. Alzubi[1] · Jafar A. Alzubi[2] · Tareq Mahmod Alzubi[3] · Ashish Singh[4]

Accepted: 27 October 2022 / Published online: 7 February 2023

Abstract
Malware refers to malicious software developed to penetrate or damage a computer system without any owner's informed consent. It uses target system susceptibilities, like bugs in legitimate software that can be harmed. For dealing with the new malware, new approaches have been developed to identify and prevent any damage caused. The recent advances in Deep Learning (DL) models are useful for malware detection because they are trained via feature learning instead of task-specific approaches. This paper presents an Optimal Encoder-Decoder Driven LSTM Networks for Malware Detection and Classification (OELSTM-MDC) technique. The presented OELSTM-MDC technique involves the identification and classification of malware. To accomplish this, the OELSTM-MDC model applies pre-processing in the initial stage for data normalization. In addition, Quantum Mayfly Optimization-based Feature Selection (QMFO-FS) approach is derived from choosing an optimal subset of features. Finally, the Butterfly Optimization Algorithm (BOA) is employed for optimal hyperparameter tuning of the ELSTM model. A wide range of empirical analysis is investigated on benchmark datasets to assess the better malware classification performance of the OELSTM-MDC model. It is also compared with the conventional machine learning models such as Random Forest, XGBoost, support vector machine, etc. According to the comparison studies, the OELSTM-MDC model outperformed conventional techniques by detecting the malware class and benign class with accuracy of 97.14% and 98.33% based on the training and testing datasets.

Keywords Security · Malware detection · Machine learning · Deep learning · LSTM · Metaheuristics · Feature selection

1 Introduction

The Cloud Computing (CC) [1] features and advantages attracted the user gradually. The server stores massive amounts of sensitive user information, which are securely shared [2] and accessed by many users. However, the remote accessing and fast sharing of data increases malware attacks and threats in data at cloud servers [3–5]. Parallelly, it also attracted malware developers and cyber attackers. Malware is an intrusive software, including spyware, trojan horse, worm, adware, ransomware, virus, etc., that primarily aim to disturb the system. It is categorized into two classes - first-generation malware and second-generation malware. The first-generation malware handles the concept in which malware structure remains unchanged. The second-generation malware changes randomly after each infection and it is very much different from each other. Each malware transactions generate a novel structure in terms of results [6, 7]. The dynamic characteristics of the malware make it hard to quarantine and detect [8, 9]. The key technology for detecting malware is heuristic-based, signature-based, Machine Learning (ML), Deep Learning (DL) Multi-attribute decision-making [10, 11], and normalization [12].

The Intrusion detection approach is one of the most promising technology used to detect malware in the cloud network. It is not only applied in the cloud network but also used in the spatially distributed sensors, and many other fields, which is an important research area investigated in [13–19]. Security analysts and researchers must continually enhance the malware detection system in which endpoint detection and protection are top priorities [20, 21]. Endpoint protection offers a set of security programs involving sandboxing firewalls, anti-spam, URL filtering, and email protection. Especially anti-malware software offers the final layer of defence. There are two

✉ Jafar A. Alzubi
j.zubi@bau.edu.jo

Extended author information available on the last page of the article.

kinds of analysis available, namely static and dynamic. The static analysis comprises inspecting an executable without implementation [22]. These two kinds of analysis have limitations and advantages and complement one another. Conventional malware analysis and detection cannot keep pace with variants and new attacks. Organizations are experiencing the serious problem of handling millions of attacks. Additionally, the organization faces a lack of cybersecurity talent and skills [23]. The recognized issue presents a great opportunity for ML to change and impact the cybersecurity landscape considerably. It is because of its capability to deal with the massive number of information [24, 25].

This paper presents an Optimal Encoder-Decoder Driven LSTM Networks for Malware Detection and Classification (OELSTM-MDC) technique. The presented OELSTM-MDC technique applies to pre-processing in the initial stage for data normalization. The Quantum Mayfly Optimization-based Feature Selection (QMFO-FS) technique is derived from selecting an optimal subset of features. Furthermore, the ELSTM classification model is applied to identify and classify malware. Lastly, the Butterfly Optimization Algorithm (BOA) enhances malware detection and classification performance. Moreover, Table 1 represents the acronyms used in the proposed malware detection and classification model.

1.1 Motivation

Most of the research works have employed various ML and DL models for efficient and secure malware detection and classification. However, as per the literature [26–29] associated with the malware detection approaches, the researchers/authors have not tackled the security and privacy issues that can arise while performing malware detection. Due to this, any malicious attacker can forge a system. Thus, there is a need to design a secure and efficient malware detection and classification model with higher accuracy. To resolve the challenges mentioned earlier, we have proposed a quantum mayfly optimization with encoder-decoder-driven LSTM networks for malware detection and classification with higher efficiency and accuracy than the conventional approaches.

Table 1 List of acronyms

Acronym	Definition
DL	Deep learning
LSTM	Long short-term memory
OELSTM-MDC	Optimal Encoder-Decoder Driven LSTM Networks for Malware Detection and Classification
QMFO-FS	Quantum Mayfly Optimization-Selection based Feature
BOA	Butterfly optimization algorithm
CC	Cloud computing
ML	Machine learning
CNN	Convolutional Neural Network
ML	Machine learning
MFs	Mayflies
QC	Quantum computing
RNN	Recurrent Neural Network

1.2 Contributions

The research contributions are summarized as follows:

- We propose a quantum mayfly optimization with encoder-decoder-driven LSTM networks for malware detection and classification. It mainly consists of the QMFO-FS technique for the initial selection of features that can be used for classifying and identifying using the ELSTM technique.
- Furthermore, the BOA algorithm is applied to strengthen the performance of malware detection and classification based on the malware and benign class.
- Finally, the performance of the proposed system has been simulated with the applied OELSTM-DC model. The results yield an accuracy of 97.14% and 98.33% based on the malware and benign class in the training and testing dataset.

2 Related work

Most of the cybersecurity and malware detection solutions say AI-powered antimalware tools efficiently detect modern malware attacks. Research has projected different techniques and learning (ML and DL) technologies [30, 31] for malware detection. The ML technique can derive a classification from limited training instances. Thus, this technique prevents the need to determine signatures explicitly in emerging malware detectors. In previous years, the ML method has triggered a radical shift in several fields, including cyber-security. Over the decade, anti-malware communities and researchers have reported many ML and DL-based models to develop malware detection and analysis schemes.

Fournier et al. [32] implemented and designed an architecture for detecting malware on Android devices to protect financial and private data for the mobile application of the ATISCOM project. Then, they gradually enhanced the presented method for the recently installed application on an Android device. The researchers in [33] presented AdMat - an efficient architecture for characterizing Android

applications by processing them as images. The innovation of the study lies in constructing an adjacent matrix for all the applications. This matrix acts as an "input image" to the CNN, allowing them to learn to differentiate between benign and malicious applications and malware families. Damaševičius et al. [34] presented an ensemble classification-based method for detecting malware. A CNN and stacked ensemble of dense can implement the classification.

In [35], a DL-based method is proposed to categorize malware variants according to a hybrid mechanism. The major objective is to present a hybrid structure that incorporates two extensive pre-trained network systems in an enhanced way. This structure comprises four major phases: training the proposed deep neural network architecture, data acquisition, evaluation of the trained deep neural network, and designing deep neural network architecture. The researchers in [36] proposed a malware detection technique based on a supervised ML algorithm. They performed a static analysis of the data extracted from the Drebin dataset. They provided a brief review of other studies in the field. Next, estimate six common classification methods under distinct configurations in terms of i) feature selection and ii) capacity to detect Android malware.

Marin et al. [37] examine the DL techniques on certain problems of classification and detection of malware. They considered raw measurement directly coming from the stream of monitored bytes as input to the presented method. A DL technique, DeepMAL can capture the fundamental statistics of malicious traffic without expert hand-crafted features. It estimates distinct raw-traffic feature representations, including flow-level and packet ones.

Later, Agarkar et al. [26] discussed a malware detection and classification model using machine learning to address the behaviour-based detection methods for malware detection. Then, Eboya et al. [27] investigated a malware detection framework for the IoT ecosystem. However, the authors in [27] did not consider the performance issues of detection. The researchers in [28] introduced a malware classification approach based on the VGG19 network. The authors in [29] discussed a client-server malware detection model utilizing machine learning for android applications. To improve the accuracy of the malware detection system in [38], the authors proposed a flood attacks-based protection model for complex networks.

Manickam et al. [39] presented DDoS attacks-based dataset based on Internet Control Message Protocol with higher detection accuracy and precision. The authors in [40] discussed an efficient method for fine-grained tasks in edge computing along with optimized energy usage. However, as mentioned earlier, the researchers need to focus on the security and privacy issues in the malware detection and classification approaches. Additionally, some of the research works did not consider the accuracy and precision parameters that decide the performance of a malware detection system. Therefore, to meet the mentioned challenges, we have proposed a preserved quantum mayfly optimization with encoder-decoder networks for malware detection and classification with higher accuracy and efficiency. Table 2 presents the comparative analysis of various state-of-the-art malware detection and classification approaches with the proposed system.

3 Proposed model

In this paper, a new OELSTM-MDC algorithm is introduced for the identification and classification of malware. The presented OELSTM-MDC technique undergoes a series of sub-processes: pre-processing, QMFO-based feature subset selection, ELSTM classification, and BOA-based hyperparameter optimization. The utilization of BOA helps to significantly enhance the overall malware detection performance of the ELSTM model. The entire block diagram of the OELSTM-MDC approach is shown in Fig. 1

A dataset has been developed that further utilises a feature selection using the QMFO technique. Initially, the data is being collected considering the training and testing dataset, including malware and benign class. Furthermore, the training dataset is pre-processed to remove the missing and null values. Then, OELSTM-DC is applied to classify the training dataset based on the malware and benign class for malware detection. Further, parameter tuning is performed using BOA to enhance the performance of malware detection in terms of efficiency and accuracy.

3.1 Pre-processing

Androguard was a complete package tool infrastructure to interrelate with Android files and has restricted only to the python environment. It could be employed as a tool for reversing engineering single Android applications. Such classification could be vital to select features which require the class a new record is going to. The permission and API calls are removed from all Android applications and integrated as a limited feature in the data set. Thus, a data frame contains a feature (column) and application (row). Every column indicates the specific permission or API call with a binary value. However, rows validate the group of malware and benign APK files. Table 3 shows the used parameters and symbols in the proposed system.

Table 2 Comparative analysis of state-of-the-art malware detection and classification approaches with the proposed system

Author	Year	Objective	Pros	Cons
Agarkar et al. [26]	2020	Proposed a behaviour-based malware detection and classification using machine learning	Improved accuracy	Need to implement in a dynamic environment
Eboya et al. [27]	2020	Discussed a Malware detection framework for the IoT ecosystem	Improved anomaly detection and accuracy	Less effort to tackle security and privacy issues
Awan et al. [28]	2021	Investigated a malware classification approach based on the VGG19 network	High performance	Security issues against malicious attack
Fournier et al. [29]	2021	Presented a client-server malware detection framework using machine learning for android applications	Enhanced correlation coefficient and minimum computation time	Security challenges against DDoS, man-in-the-middle attacks
Vu et al. [41]	2021	Discussed a CNN-based android malware detection framework	Covered dynamic malware families	No discussion on malicious attacks
Khalaf et al. [38]	2021	Designed a flooding attacks-based protection model for complex networks	Improved accuracy and precision, secure against DDoS attacks	Need to implement in real-time
Manickam et al. [39]	2022	Proposed a DDoS attacks dataset based on Internet Control Message Protocol	High detection accuracy	Need to consider a single point of failure and man-in-the-middle attacks
Lakhan et al. [40]	2022	Discussed an energy-efficient method for fine-grained tasks in edge computing	Optimized energy usage	Less effort on accuracy and efficiency
The proposed system	2022	Quantum mayfly optimization with encoder-decoder driven LSTM networks for malware detection and classification model	High accuracy, efficiency, and security	

3.2 Process involved in QMFO-FS technique

The MO technique is presented by Zervoudakis and Tsafarakis [42]. They simulate the mating procedure demonstrated by mayflies (MFs) in nature. In MO technique work primarily by creating two arbitrary population sets demonstrating the female and male sets correspondingly. All the MFs placed from the problem space implies the potential solution to a problem. The place has been demonstrated as $ddimensionalvector = (x_1, x_2, ..., x_n)$, and $f(x)$ is the main function to evaluate the performance of all MFs. The MFs place alters their velocity $v = (v_1, v_2, ..., v_n)$. However, the flying direction of all MFs is defined as the optimum individual flying experiences of all the MFs (pbest) and optimum swarm social flying experiences (gbest). As the male moved in a swarm and danced on some water meters, it could not move at maximum speed. Therefore, the velocity of male MF has been calculated with the help of Eq. 1.

$$v_{ij}^{t+1} = v_{ij}^{t} + a_2 e^{-\beta r_p^2}(pbest_{ij} - x_{ij}^{t}) + a_1 e^{-\beta r_g^2}(gbest_{ij} - x_{ij}^{t}) \quad (1)$$

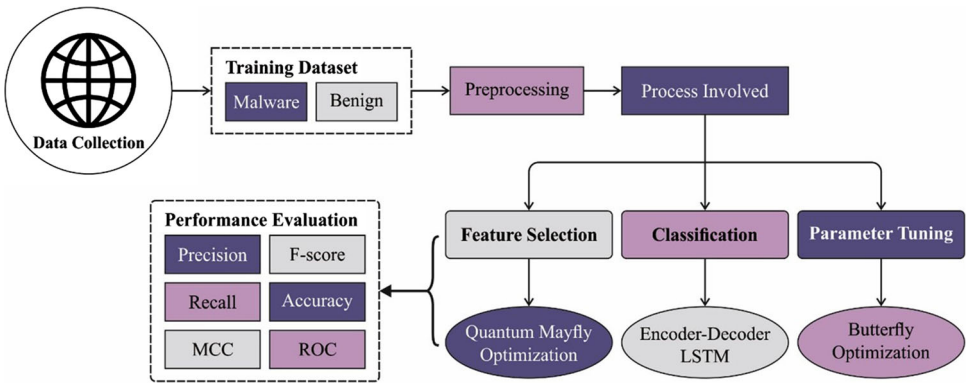

Fig. 1 Block diagram of OELSTM-MDC technique

Table 3 Proposed system parameters

Constants/Parameters	Definition	Constants/Parameters	Definition
a1	Constant for scaling the contribution of social and cognitive elements	d	Coefficient of nuptial dances
v_{ij}^t	Male and female MF velocity	r_{mf}	Distance between male and female MFs
J	MF number	fl	Random walk coefficient
T	Time step	N	Count of male MFs
x_i^t	Present place of male	β	Visibility coefficient
y_i^t	Present place of female	a2	Constant for scaling the contribution of social and cognitive elements
$pbest_j$	Optimum place of MF	male/female	Male parent, female parent

whereas v_{ij}^t refers the male MF velocity, $x_{i_j}^t$ indicates the place, j implies the MF number, $j = 1, ..., n$ represents the space dimensional, t denotes the time step. However, a_1 and a_2 are constants executed for corresponding constants to scale the contribution of social and cognitive elements. Also, $pbest_j$ represents the optimum place stayed by MF i and N defines the count of male MFs. Lastly, β represents the visibility co-efficient that limits the visibility of MFs to other MFs, but r_p and r_g indicate the distances amongst x_j and $pbest_i$ and $gbest$ correspondingly. A novel place of the male is computed as adding the velocity v_i^{t+1} to the present place. It is represented by Eq. 2.

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2}$$

An optimum MFs endure for executing its nuptial dance. So, the optimum MFs have altered their velocity based on the subsequent relation, represented by Eq. 3.

$$v_i^{t+1} = v_{ij}^t + d * r \tag{3}$$

d refers to the co-efficient of nuptial dances and r denotes the arbitrary number between the range of -1 and 1. These movements present a stochastic element to this technique. The velocity of females is computed with the help of Eq. 4.

$$v_i^{t+1} = \begin{cases} v_{ij}^t + a_3 e^{-\beta r_{mf}^2}(x_{ij}^t| - y_{ij}^t), & iff(y_i) > f(x_i) \\ v_{ij}^t + fl * r, & iff(y_i) \leq f(x_i) \end{cases} \tag{4}$$

whereas v_{ij}^t indicates the female MF velocity, y_{ij}^t refers the place, i is MF number, $j = 1, ..., n$ indicates the space dimensional, t denotes the time step. In addition, a_3 is a constant executed for scaling the contribution of social and cognitive elements. However, β represents the visibility co-efficient, but r_{mf} refers to the distance between female and male MFs.

At last, fl represents the random walk co-efficient executed in case of attraction between a female and male failed, and r stands for the arbitrary number with -1 and 1 range. A novel place of female MF was calculated as added velocity v_i^{t+1} to the present place. It is represented by Eq. 5.

$$y_i^{t+1} = y_i^t + v_i^{t+1} \tag{5}$$

The mating procedure amongst MFs is executed with the crossover operator. As stated previously, fitness value has been utilized for selecting the parent to mate, and outcomes in two offspring are created with the help of Eqs. 6 and 7.

$$offspring1 = L * male + (1 - L) * female \tag{6}$$

$$offspring2 = L * female + (1 - L) * male \tag{7}$$

In these equations, male refers to the male parent, female indicates the female parent, and L stands for the arbitrary number in an existing range. A primary velocity $offspring1$ and $offspring2$ are considered that zero.

The QMFO algorithm has been developed by utilising Quantum Computing (QC) concepts to improve the outcomes of the MFO algorithm. It is a new type of computing model based on quantum theory, such as quantum entanglement, quantum measurement, and state superposition, which adapt the model. The core component of QC is qubit [43]. The two fundamental states $|0>$ and $|1>$ form a qubit, represented by Eq. 8 as a linear integration of both states.

$$|Q> = \alpha|0> + \beta|1> \tag{8}$$

$|\alpha|^2$ denotes the probability of observing state $|0>$, $|\beta|^2$ indicator the probability of observing state $|1>$, where $|\alpha^2| + |\beta^2| = 1$. The Quantum is composed of n qubits. According to the nature of quantum superposition,

every Quantum comprises 2^n possible values. An n-qubits quantum is represented by Eq. 9.

$$\Psi = \sum_{x=0}^{2^n-1} C_x|x>, \sum_{x=0}^{2^n-1} |C_x|^2 = 1 \tag{9}$$

Quantum gate changes the state of qubits, namely NOT gate, rotation gate, Hadamard gate, etc. The rotation gate is determined as a mutation operator for improving the quanta method and finding the global optimum solution.

The rotation gate can be defined by Eqs. 10 and 11.

$$\begin{bmatrix}\alpha^2(t+1)\\ \beta^2(t+1)\end{bmatrix} = \begin{bmatrix}\cos(\Delta\theta^d) & -\sin(\Delta\theta^d)\\ \sin(\Delta\theta^d) & \cos(\Delta\theta^d)\end{bmatrix}\begin{bmatrix}\alpha^d(t)\\ \beta^d(t)\end{bmatrix}, d = 1, 2, ..., n \tag{10}$$

$$\Delta\theta^d = \Delta \times S(\alpha^d, \beta^d) \tag{11}$$

$\Delta\theta^d$ indicates the rotation angle of the qubit, whereby Δ and $S(\alpha^d, \beta^d)$ denote the size and direction of rotation correspondingly.

3.3 Steps involved in ELSTM-based classification

Once the feature subsets are elected, the ELSTM model is utilized to classify the malware. Traditional Recurrent Neural Network (RNN) utilizes preceding context states to determine future states. Bidirectional RNN (BRNN) processes data in two directions with two different hidden states later propagated towards a similar output layer [44]. BRNN employs two RNNs to assist with backward and forward data regarding the sequence at each time step. BRNN calculates the output sequence y, the hidden forward sequence h_f and the backward hidden sequence h_b by iterating data from the backward layer $t = T$ to $t = 1$. Next, data in the other networks are propagated from $t = 1$ to $t = T$ for updating the output layer; once these two networks are integrated, data is propagated bi-directionally. A bi-directional LSTM network (BLSTM) was initially developed for word embedding in NLP for accessing long-range contexts or states in two directions. BLSTM network has been utilized in real-time sequence processing problems, including speech synthesis, phoneme classification, and continuous speech recognition. Figure 2 illustrates the LSTM network structure.

ELSTM is a sequence-to-sequence method for mapping a set length input to a set length output. Therefore, the input sequence of video frames $(x_1, ..., x_n)$, and the output can be the sequence $X(t+1)X(t+2)X(t+3)X(t+m)$ of words $(y_1, ..., y_m)$. Thus, evaluate the conditional possibility of output sequence $(y_1, ..., y_m)$, assumed sequence of input $(x_1, ..., x_n)$, that is $p(y_1, ..., y_m|x_1, ..., x_n)$. In multi-step sequence predicting, the input and output are of parameter length. In the encoding stage, assuming a sequence of input, the ELSTM calculates a sequential hidden layer.

3.4 Optimal hyperparameter tuning using BOA

The BOA has been applied to modify the hyperparameter values in the ELSTM model. The presented BOA replicates the performance of butterflies (BFs) on mating and food source finding [45]. This technique employs two distinct navigation patterns for searching the field. During the exploration stage ($r_1 \leq p$), BF move towards the optimal BF of the colony, whereas from the exploitation stage ($r_1 > p$), BF performs an arbitrary search inside the searching space by moving toward an arbitrary BF from the colony. Eq. 12, arithmetically expresses this two-searching pattern.

$$\begin{cases} if r_1 \leq p \; {}^{t+1}X_i = {}^tX_i(r_2^2 \times g^* - {}^tX_i) \times \phi_i & Global\ Search \\ if r_1 > p \; {}^{t+1}X_i = {}^tX_i(r_3^2 \times {}^tX_i - {}^tX_k) \times \phi_i & Local\ Search \end{cases} \tag{12}$$

Whereas t and $t+1$ indicate the present and upgraded state of the respective parameter. The position of the optimal BF in the colony is represented as g^*. t_{X_j} and t_{X_k} denotes the location of two randomly chosen BFs. r_1, r_2 and r_3 indicates three arbitrary values within [0, 1]. ϕ_i denotes the fragrance factor. It is determined by Eq. 13.

$$\phi_i = cI^a \tag{13}$$

ϕ_i denotes the fragrance magnitude to i^{th} BF; I and a indicate the intensity of the stimulus and the fluctuating absorption degree, and c represents a coefficient. I refers the related intensity to the main function value, and i^{th} BF can be assumed as $f(X_i)$, whereas f returns objective function value. The a and c coefficients are chosen from the

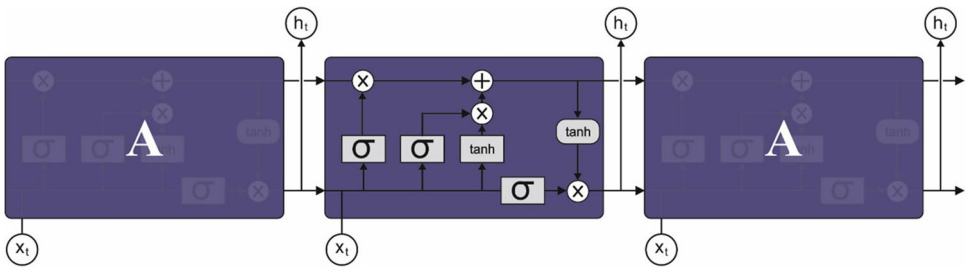

Fig. 2 LSTM networks

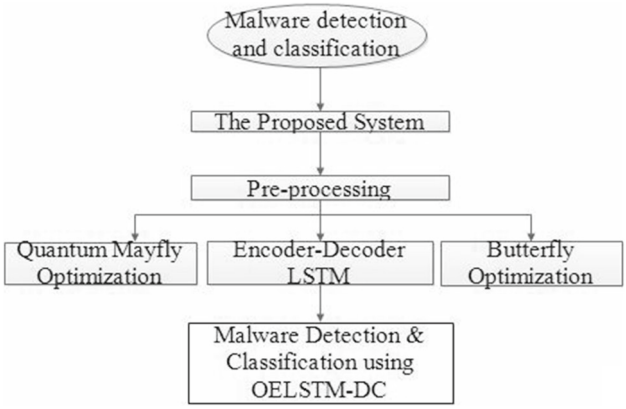

Fig. 3 Sequence flow of the proposed system

range of [0, 1]; p denotes the probability switch that defines the searching behaviour. The BOA approach constructs a fitness function to increase classification performance. It calculates a positive integer to indicate the candidate solutions' improved performance. The best solution has the lowest error rate, whereas the worst option has a higher error rate. This work's fitness function minimises the classification error rate, as shown in Eq. 14.

$$fitness(x_i) = Classifier\,Error\,Rate(x_i) = \frac{number\ of\ misclassified\ samples}{Total\ number\ of\ samples} \times 100 \quad (14)$$

Figure 3 presents the sequence flow of the proposed system. It initiates with the pre-processing, feature selection and selection using different classification techniques. Now, it is further used for malware detection and classification with the OELSTM-DC model.

Table 4 Model parameters

Parameter	Value
Epochs	100
Batch Size	90
Optimizer	Adam
Activation	ReLU
Validation Spilt	0.1

4 Experimental validation

In this section, malware detection and classification performance have been evaluated.

4.1 Dataset description

The dataset involves 2000 applications in which 1000 applications fall under the category of malicious class and another 1000 applications fall under the category of benign class. The considered dataset includes various features. We must select the relevant one based on the malware detection and classification requirement. The dataset's relevant features include 75 features in which classification using the QMFO-FS technique requires 25 features for malware detection based on the malicious and benign classes.

4.2 Malware detection and classification using OELSTM-MDC technique

In this section, the malware detection and classification outcomes of the OELSTM-MDC model are tested using dataset [46, 47]. The dataset consists of 2000 applications divided equally into two classes (malicious and benign),

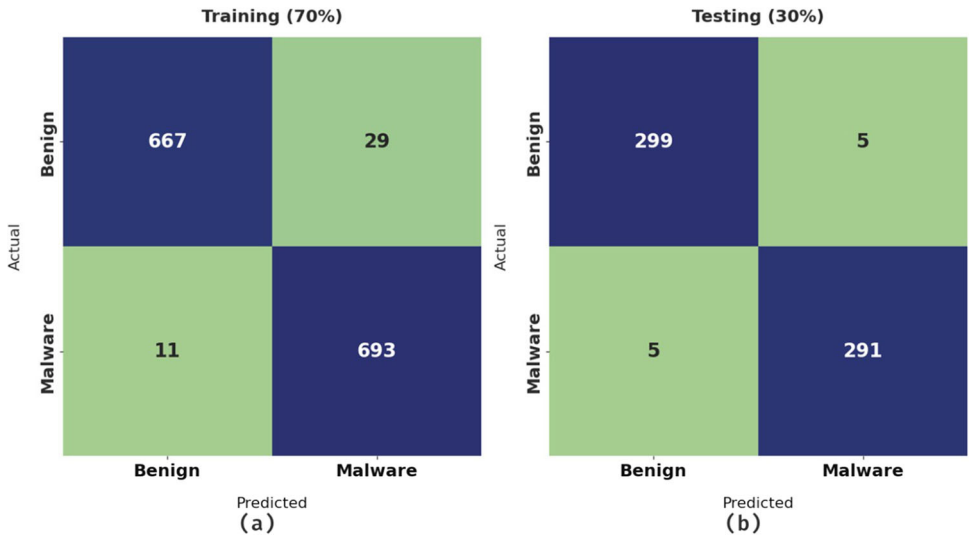

Fig. 4 Confusion of OELSTM-MDC technique under the training of 70% and testing of 30%

Table 5 Result analysis of OELSTM-MDC technique on the training of 70% and testing of 30%

Class Label	Accuracy	Precision	Recall	F-Score	MCC
Training (70%)					
Benign	97.14	98.38	95.83	97.09	94.32
Malware	97.14	95.98	98.44	97.19	94.32
Average	**97.14**	**97.18**	**97.14**	**97.14**	**94.32**
Testing (30%)					
Benign	98.33	98.36	98.36	98.36	96.67
Malware	98.33	98.31	98.31	98.31	96.67
Average	**98.33**	**98.33**	**98.33**	**98.33**	**96.67**

with 1000 applications in each class. The dataset includes 75 features and the QMFO-FS technique has chosen a set of 25 features.

Figure 4 exhibits the confusion matrices generated by the OELSTM-MDC model on 70% of training and 30% of testing datasets. Figure 4a illustrates that the OELSTM-MDC model has effectually recognized 667 samples under the benign class and 693 samples under the malware class. In addition, Fig. 4b shows that the OELSTM-MDC technique has effectually recognized 299 samples under the benign class and 291 samples under the malware class. Table 4 shows the parameters used to implement the OELSTM-DC model considering the dataset for malware detection and classification.

Table 5 reports a brief malware classification outcome of the OELSTM-MDC model on training data of 70% and testing data of 30%.

Figure 5 offers detailed classifier outcomes of the OELSTM-MDC model on the training dataset. The OELSTM-MDC model has classified the samples under benign class with $accu_y$, $prec_n$, $reca_l$, F_score, and

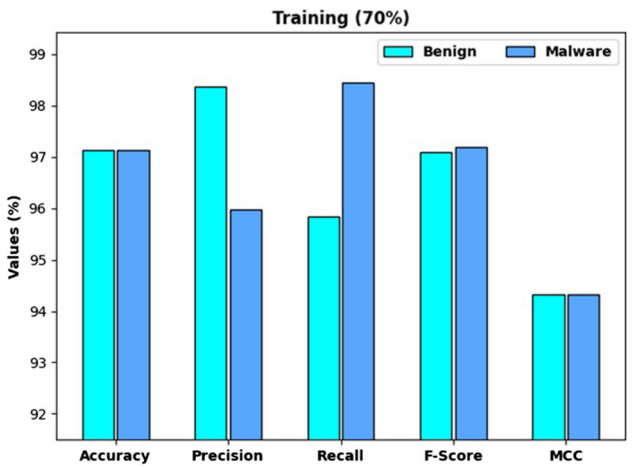

Fig. 5 Result analysis of OELSTM-MDC technique on the training of 70% dataset

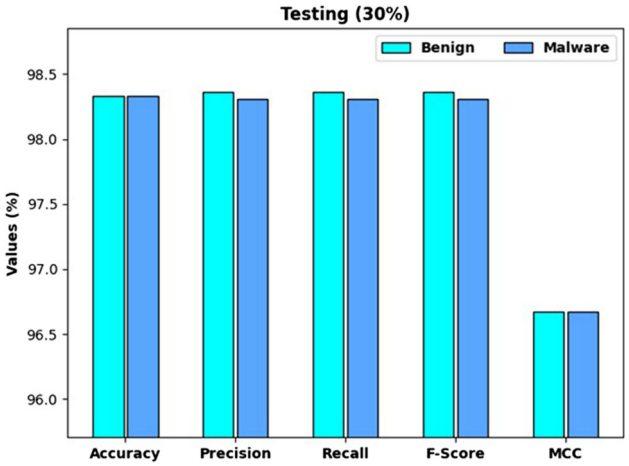

Fig. 6 Result analysis of OELSTM-MDC technique on testing of 30% dataset

MCC of 97.14%, 98.38%, 95.83%, 97.09%, and 94.32% respectively. Moreover, the OELSTM-MDC technique has classified the samples under the Malware class with $accu_y$, $prec_n$, $reca_l$, F_score, and MCC of 97.14%, 95.98%, 98.44%, 97.19%, and 94.32% correspondingly.

Figure 6 provides detailed classifier outcomes of the OELSTM-MDC model on the testing dataset. The OELSTM-MDC technique has classified the samples under benign class with $accu_y$, $prec_n$, $reca_l$, F_score, and MCC of 98.33%, 98.36%, 98.36%, 98.36%, and 96.67% respectively. Additionally, the OELSTM-MDC model has classified the samples under the Malware class with $accu_y$, $prec_n$, $reca_l$, F_score, and MCC of 98.33%, 98.31%, 98.31%, 98.31%, and 96.67% correspondingly.

Figure 7 showcases the classifier results of the ODCNN-RFIC method on the test dataset. Figure 7a depicts that the ODCNN-RFIC technique has showcased effective precision-recall outcomes under the training of 70%. At the same time, Fig. 7b depicts that the ODCNN-RFIC technique has showcased effective precision-recall outcomes under testing of 30%. In addition, Fig. 7c illustrates that the ODCNN-RFIC technique has offered ROC, resulting in a maximum training performance of 70%. Also, Fig. 7d illustrates that the ODCNN-RFIC technique has offered ROC, resulting in maximum performance on testing of 30%.

Figure 8 depicts the overall accuracy of the OELSTM-MDC system's results analysis on the test data. The results exhibited that the OELSTM-MDC approach has achieved improved validation accuracy compared to training accuracy. It is also worth noting that the accuracy values get saturated as the number of epochs increases.

Figure 9 shows the total loss outcome analysis of the OELSTM-MDC technique on the test data. The figure revealed that the OELSTM-MDC technique had denoted the reduced validation loss over the training loss. Furthermore,

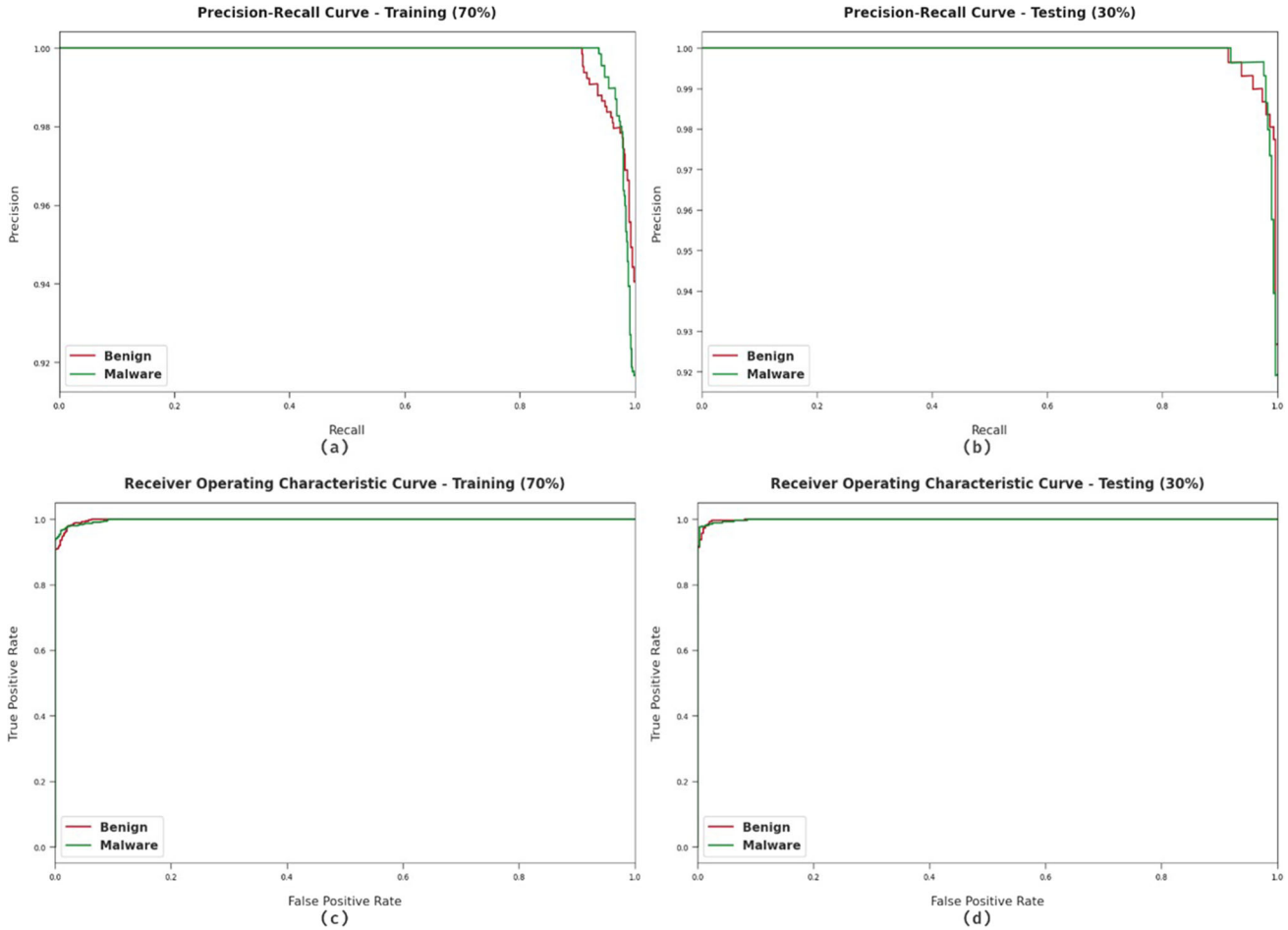

Fig. 7 a) Precision recall-training 70%, b) Precision recall-testing at 30%, c) ROC-training at 70%, d) ROC-testing at 30%

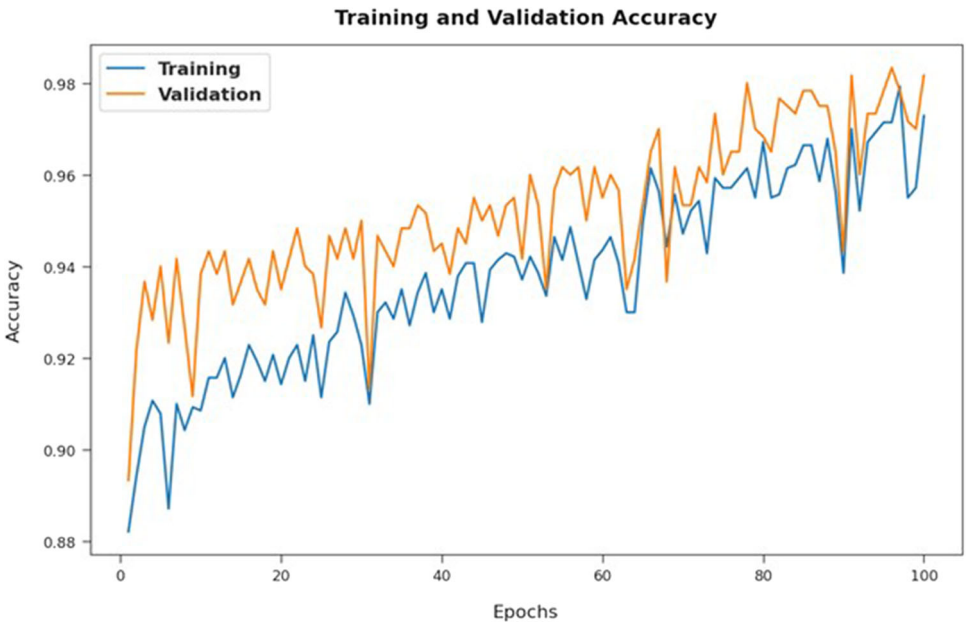

Fig. 8 Accuracy analysis of OELSTM-MDC technique

Fig. 9 Loss analysis of OELSTM-MDC technique

the loss values become saturated as the number of epochs increases.

A comparison study with existing models is made in Table 6 [48] to ensure the OELSTM-MDC model's better outcomes.

Figure 10 reports a comparative $prec_n$ investigation of the OELSTM-MDC model with recent models. The results indicated that the GA-SVM and LR-MLP models had lowered $prec_n$ values of 94.93% and 94.95%, respectively. The IG-Random Forest and RST-PSO algorithms have slightly increased $prec_n$ values of 95.68% and 95.62%, respectively. Moreover, the CFS-Random Forest model has accomplished a reasonably $prec_n$ of 96.96%. Though the RDT-XGBoost and E-LSTM models have exhibited considerable $prec_n$ of 97.58% and 97.85%, the OELSTM-MDC model has depicted a maximum $prec_n$ of 98.33%.

Figure 11 defines a comparative $reca_l$ examination of the OELSTM-MDC model with recent models. The results revealed that the GA-SVM and LR-MLP models have resulted in lower $reca_l$ values of 94.37% and 95.76%, respectively. Similarly, the IG-Random Forest and RST-PSO algorithms have slightly increased $reca_l$ values of 94.60% and 94.10%, respectively. Besides, the CFS-Random Forest approach has accomplished a reasonably $reca_l$ of 96.28%. Lastly, the RDT-XGBoost and E-LSTM methodologies have exhibited considerable $reca_l$ of 97.30% and 97.38%. The OELSTM-MDC technique has depicted a maximum $reca_l$ of 98.33%.

Figure 12 showcases a comparative Acc_y analysis of the OELSTM-MDC model with recent models. The results revealed that the GA-SVM and LR-MLP models have resulted in lower Acc_y values of 95.07% and 96.17%, respectively. At the same time, the IG-Random Forest and

Table 6 Comparative analysis of OELSTM-MDC algorithm with existing methods

Method	Precision	Recall	Accuracy	F-Score
CFS-Random Forest	96.96	96.28	95.00	95.86
RDT-XGBoost	97.58	97.30	95.71	94.16
IG-Random Forest	95.68	94.60	97.35	97.01
IG-Random Forest	95.68	94.60	97.35	97.01
GA-SVM	94.93	94.37	95.07	97.99
RST-PSO	95.62	94.10	97.45	91.20
LR-MLP	94.95	95.76	96.17	96.00
E-LSTM	97.85	97.38	97.79	98.09
OELSTM-MDC	98.33	98.33	98.33	98.33

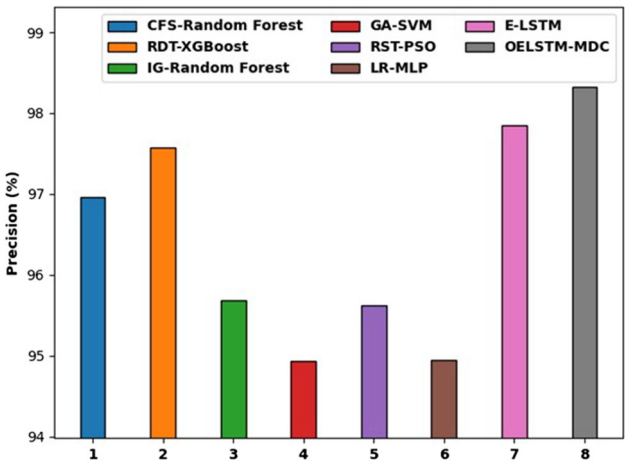

Fig. 10 Precision analysis of OELSTM-MDC technique with recent algorithms

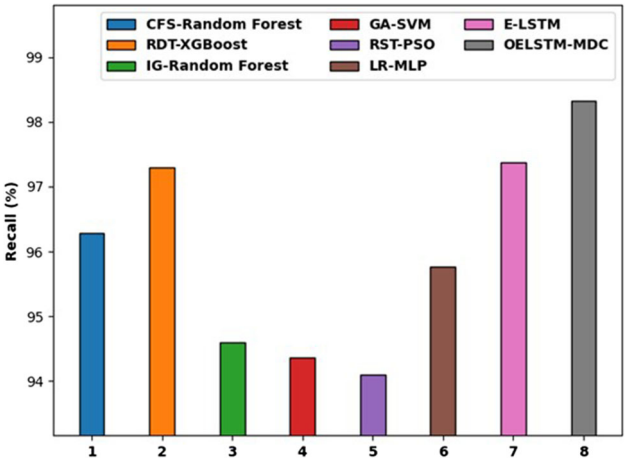

Fig. 11 Recall analysis of OELSTM-MDC technique with recent algorithms

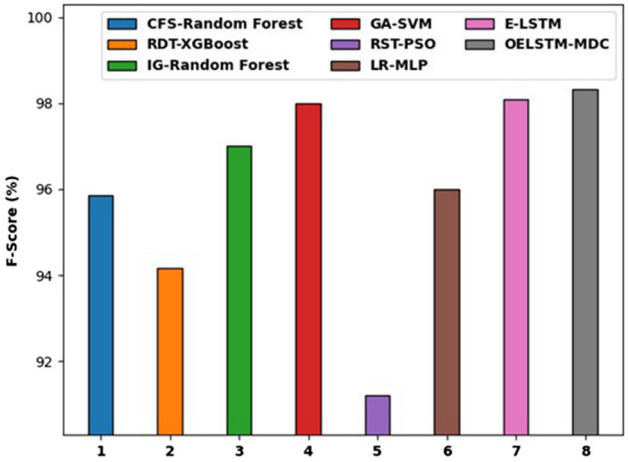

Fig. 13 F-score analysis of OELSTM-MDC technique with recent algorithms

RST-PSO algorithms have slightly increased Acc_y values of 97.35% and 97.45%, respectively. Furthermore, the CFS-Random Forest model has accomplished reasonably Acc_y of 95%. Eventually, the RDT-XGBoost and E-LSTM methods have exhibited considerable Acc_y of 95.71% and 97.79%. The OELSTM-MDC model has depicted a maximum Acc_y of 98.33%.

Figure 13 determines a comparative F_score examination of the OELSTM-MDC model with recent models. The outcomes indicated that the GA-SVM and LR-MLP models have resulted in lower F_score values of 97.99% and 96%, respectively. Also, the IG-Random Forest and RST-PSO algorithms have reached slightly increased F_score values of 97.01% and 91.20% correspondingly. In addition, the CFS-Random Forest model has accomplished a reasonably F_score of 95.86%. At last, the RDT-XGBoost and E-LSTM models have exhibited considerable F_score of 94.16% and 98.09%. The OELSTM-MDC methodology has depicted a maximum F_score of 98.33%.

The simulation results and discussion show that the OELSTM-MDC model has produced the best results compared with the recent algorithms.

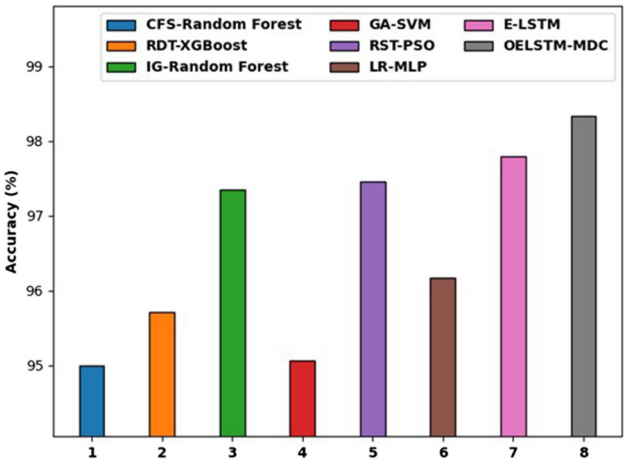

Fig. 12 Accuracy analysis of OELSTM-MDC technique with recent algorithms

5 Conclusion

This paper established a new OELSTM-MDC algorithm to identify and classify malware. The presented OELSTM-MDC technique undergoes a series of sub-processes: pre-processing, QMFO-based feature subset selection, ELSTM classifier, and BOA-based hyperparameter optimisation. The utilization of BOA helps to significantly enhance the overall malware detection performance of the ELSTM technique. A wide-ranging experimental analysis is carried out on the benchmark dataset to examine the enhanced performance of the OELSTM-MDC approach. The comparative analysis reported the improved outcomes of the OELSTM-MDC model on existing techniques. Therefore, the OELSTM-MDC approach is utilized as a proficient approach for malware classification with an accuracy of 97.14% and 98.33% based on the benign and malware class category in the training and testing datasets. In future, hybrid DL models can be applied to boost the efficiency of the OELSTM-MDC technique in a dynamic environment. Furthermore, various DL models can be explored and implemented to improve the overall performance of malware detection and classification.

Funding The authors did not receive support from any organization for this work.

Data Availability Not applicable.

Declarations

Consent for Publication For this study consent for publication is not required.

Informed Consent For this study informed consent is not required.

Competing interests There is no conflict of interest.

References

1. Singh A, Chatterjee K (2017) Cloud security issues and challenges: a survey. J Netw Comput Appl 79:88–115. https://doi.org/10.1016/j.jnca.2016.11.027
2. Gutub A (2022) Boosting image watermarking authenticity spreading secrecy from counting-based secret-sharing. CAAI Trans Intell Technol
3. Gibert D, Mateu C, Planes J (2020) The rise of machine learning for detection and classification of malware: research developments, trends and challenges. J Netw Comput Appl 153(C). https://doi.org/10.1016/j.jnca.2019.102526
4. Chakraborty A, Alam M, Dey V, Chattopadhyay A, Mukhopadhyay D (2021) A survey on adversarial attacks and defences. CAAI Transactions on Intelligence Technology 6(1):25–45
5. Shhadat I, Bataineh B, Hayajneh A, Al-Sharif ZA (2020) The use of machine learning techniques to advance the detection and classification of unknown malware. Procedia Comput Sci 170:917–922. https://doi.org/10.1016/j.procs.2020.03.110
6. Chen T, Mao Q, Yang Y, Lv M, Zhu J (2018) Tinydroid: a lightweight and efficient model for android malware detection and classification. Mob Inf Syst 2018:1–9. https://doi.org/10.1155/2018/4157156
7. Alzubi OA, Alzubi JA, Al-Zoubi AM, Hassonah MA, KÖSE U (2022) An efficient malware detection approach with feature weighting based on harris hawks optimization. Clust Comput 25:2369–2387. https://doi.org/10.1007/s10586-021-03459-1
8. Roseline SA, Geetha S, Kadry S, Nam Y (2020) Intelligent vision-based malware detection and classification using deep random forest paradigm. IEEE Access 8:206303–206324. https://doi.org/10.1109/ACCESS.2020.3036491
9. Vu LN, Jung S (2021) Admat: a cnn-on-matrix approach to android malware detection and classification. IEEE Access 9:39680–39694. https://doi.org/10.1109/ACCESS.2021.3063748
10. Mahmood T, Ali Z (2022) Prioritized muirhead mean aggregation operators under the complex single-valued neutrosophic settings and their application in multi-attribute decision-making. J Comput Cogn Eng:56–73
11. Alamleh A, Albahri O, Zaidan A, Alamoodi A, Albahri A, Zaidan B, Qahtan S, Binti Ismail AR, Malik R, Baqer M et al (2022) Multi-attribute decision-making for intrusion detection systems: a systematic review. Int J Inf Technol Decis Mak:1–48
12. Alzubi OA (2022) Quantum readout and gradient deep learning model for secure and sustainable data access in iwsn. PeerJ Comput Sci 8:983–1007. https://doi.org/10.7717/peerj-cs.983
13. Yakici E, Karatas M (2021) Solving a multi-objective heterogeneous sensor network location problem with genetic algorithm. Comput Netw 192:108041
14. Karatas M (2020) A multi-objective bi-level location problem for heterogeneous sensor networks with hub-spoke topology. Comput Netw 181:107551. https://doi.org/10.1016/j.comnet.2020.107551
15. Karatas M, Onggo BS (2019) Optimising the barrier coverage of a wireless sensor network with hub-and-spoke topology using mathematical and simulation models. Comput Oper Res 106:36–48. https://doi.org/10.1016/j.cor.2019.02.007
16. Karatas M (2018) Optimal deployment of heterogeneous sensor networks for a hybrid point and barrier coverage application. Comput Netw 132:129–144. https://doi.org/10.1016/j.comnet.2018.01.001
17. Karatas M, Onggo BS (2016) Validating an integer non-linear program optimization model of a wireless sensor network using agent-based simulation. In: 2016 winter simulation conference (WSC), pp 1340–1351. https://doi.org/10.1109/WSC.2016.7822188
18. Alzubi OA (2022) A deep learning- based frechet and dirichlet model for intrusion detection in iwsn. J Intell Fuzzy Syst 42(2):873–883. https://doi.org/10.3233/JIFS-189756
19. Chen TM, Blasco J, Alzubi JA, Alzubi OA (2014) Intrusion detection. IET 1(1):1–9. https://doi.org/10.1049/etr.2014.0007
20. Gao H, Cheng S, Zhang W (2021) Gdroid: android malware detection and classification with graph convolutional network. Comput Secur 106:102264. https://doi.org/10.1016/j.cose.2021.102264
21. Alzubi OA, Qiqieh I, Alzubi JA (2022) Fusion of deep learning based cyberattack detection and classification model for intelligent systems. Clust Comput In Press
22. Dewanje A, Kumar KA (2021) A new malware detection model using emerging machine learning algorithms. Int J Electron Inf Eng 13(1):24–32
23. Kouliaridis V, Kambourakis G (2021) A comprehensive survey on machine learning techniques for android malware detection. Information 12(5). https://doi.org/10.3390/info12050185
24. Singh J, Singh J (2021) A survey on machine learning-based malware detection in executable files. J Syst Archit 112:101861. https://doi.org/10.1016/j.sysarc.2020.101861
25. Zhao Y, Li L, Wang H, Cai H, Bissyandé TF, Klein J, Grundy J (2021) On the impact of sample duplication in machine-learning-based android malware detection 30(3). https://doi.org/10.1145/3446905
26. Choudhary S, Sharma A (2020) Malware detection & classification using machine learning. In: 2020 international conference on emerging trends in communication control and computing (ICONC3), pp 1–4. https://doi.org/10.1109/ICONC345789.2020.9117547
27. Eboya O, Juremi JB, Shahpasand M (2020) An intelligent framework for malware detection in internet of things (iot) ecosystem. In: 2020 IEEE 8th R10 humanitarian technology conference (R10-HTC), pp 1–6. https://doi.org/10.1109/R10-HTC49770.2020.9356961
28. Awan MJ, Masood OA, Mohammed MA, Yasin A, Zain AM, Damaševičius R, Abdulkareem KH (2021) Image-based malware classification using vgg19 network and spatial convolutional attention. Electronics 10(19). https://doi.org/10.3390/electronics10192444
29. Fournier A, El Khoury F, Pierre S (2021) A client/server malware detection model based on machine learning for android devices. IoT 2(3):355–374. https://doi.org/10.3390/iot2030019
30. Chen Z (2022) Research on internet security situation awareness prediction technology based on improved rbf neural network algorithm. J Comput Cogn Eng
31. Wani A, Khaliq R (2021) Sdn-based intrusion detection system for iot using deep learning classifier (idsiot-sdl). CAAI Trans Intell Technol 6(3):281–290

32. Fournier A, El Khoury F, Pierre S (2021) A client/server malware detection model based on machine learning for android devices. IoT 2(3):355–374. https://doi.org/10.3390/iot2030019
33. Vu LN, Jung S (2021) Admat: a cnn-on-matrix approach to android malware detection and classification. IEEE Access 9:39680–39694. https://doi.org/10.1109/ACCESS.2021.3063748
34. Damaševičius R, Venčkauskas A, Toldinas J, Grigaliūnas Š (2021) Ensemble-based classification using neural networks and machine learning models for windows pe malware detection. Electronics 10(4):485
35. Aslan O, Yilmaz AA (2021) A new malware classification framework based on deep learning algorithms. IEEE Access 9:87936–87951. https://doi.org/10.1109/ACCESS.2021.3089586
36. Syrris V, Geneiatakis D (2021) On machine learning effectiveness for malware detection in android os using static analysis data. J Inf Secur Appl 59:102794. https://doi.org/10.1016/j.jisa.2021.102794
37. Marín G, Caasas P, Capdehourat G (2021) Deepmal - deep learning models for malware traffic detection and classification. In: Haber P, Lampoltshammer T, Mayr M, Plankensteiner K (eds) Data science – analytics and applications. Springer, pp 105–112
38. Khalaf B, Mostafa S, Mustapha A, Mohammed M, Mahmoud M, Al-Rimy B, Abd Razak S, Elhoseny M, Marks A (2021) An adaptive protection of flooding attacks model for complex network environments. Secur Commun Netw 2021:1–17. https://doi.org/10.1155/2021/5542919
39. Manickam S, Bdair A, Abdullah R, Alyasseri Z, Abdulkareem K, Mohammed M, Alani A (2022) Labelled dataset on distributed denial-of-service (ddos) attacks based on internet control message protocol version 6 (icmpv6). Wirel Commun Mob Comput 2022. https://doi.org/10.1155/2022/8060333
40. Lakhan A, Mohammed M, Rashid A, Kadry S, Abdulkareem K (2022) Deadline aware and energy-efficient scheduling algorithm for fine-grained tasks in mobile edge computing. Int J Web Grid Serv 18:168. https://doi.org/10.1504/IJWGS.2022.121935
41. Vu LN, Jung S (2021) Admat: a cnn-on-matrix approach to android malware detection and classification. IEEE Access 9:39680–39694. https://doi.org/10.1109/ACCESS.2021.3063748
42. Zervoudakis K, Tsafarakis S (2020) A mayfly optimization algorithm. Comput Ind Eng 145:106559. https://doi.org/10.1016/j.cie.2020.106559
43. Singh P, Huang Y-P (2019) A new hybrid time series forecasting model based on the neutrosophic set and quantum optimization algorithm. Comput Ind 111:121–139. https://doi.org/10.1016/j.compind.2019.06.004
44. Chandra R, Goyal S, Gupta R (2021) Evaluation of deep learning models for multi-step ahead time series prediction. IEEE Access 9:83105–83123. https://doi.org/10.1109/ACCESS.2021.3085085
45. Arora S, Singh S (2019) Butterfly optimization algorithm: a novel app roach for global optimization. Soft Comput 23. https://doi.org/10.1007/s00500-018-3102-4
46. APKPure (2022) DataSet_v1.0_APKPure.com.apk. https://m.apkpure.com/dataset/com.srinivasanand.dataset/download. Accessed 30 June 2022
47. Wei F, Li Y, Roy S, Ou X, Zhou W (2017) Deep ground truth analysis of current android malware. In: International conference on detection of intrusions and malware, and vulnerability assessment. Springer, pp 252–276
48. Şahin D, Kural O, Akleylek S, Kilic E (2021) A novel permission-based android malware detection system using feature selection based on linear regression. Neural Comput Appl:1–16. https://doi.org/10.1007/s00521-021-05875-1

Affiliations

Omar A. Alzubi[1] · Jafar A. Alzubi[2] · Tareq Mahmod Alzubi[3] · Ashish Singh[4]

✉ Ashish Singh
ashish.singhfcs@kiit.ac.in

Omar A. Alzubi
o.zubi@bau.edu.jo

Tareq Mahmod Alzubi
tareqzubi@bau.edu.jo

1 Prince Abdullah bin Ghazi Faculty of Information and Communication Technology, Al-Balqa Applied University, Al-Salt, Jordan

2 Faculty of Engineering, Al-Balqa Applied University, Al-Salt, Jordan

3 Prince Abdullah bin Ghazi Faculty of Information and Communication Technology, Al-Balqa Applied University, Al-Salt, Jordan

4 School of Computer Engineering, Kalinga Institute of Industrial Technology, Deemed to be University, Bhubaneswar, India