Dwarf Mongoose Optimization with Machine-Learning-Driven Ransomware Detection in Internet of Things Environment

Khalid A. Alissa 1, Dalia H. Elkamchouchi 2, Khaled Tarmissi 3, Ayman Yafoz 4, Raed Alsini 4*, Omar Alghushairy 5, Abdullah Mohamed 6 and Mesfer Al Duhayyim 7,*

1 SAUDI ARAMCO Cybersecurity Chair, Networks and Communications Department, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia
2 Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
3 Department of Computer Sciences, College of Computing and Information System, Umm Al-Qura University, Mecca 24382, Saudi Arabia
4 Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 2254, Saudi Arabia
5 Department of Information Systems and Technology, College of Computer Science and Engineering, University of Jeddah, Jeddah 21589, Saudi Arabia
6 Research Centre, Future University in Egypt, New Cairo 11845, Egypt
7 Department of Computer Science, College of Sciences and Humanities-Aflaj, Prince Sattam bin Abdulaziz University, Al-Kharj 16278, Saudi Arabia
* Correspondence: m.alduhayyim@psau.edu.sa

Abstract: The internet of things (ransomware refers to a type of malware) is the concept of connecting devices and objects of all types on the internet. IoT cybersecurity is the task of protecting ecosystems and IoT gadgets from cyber threats. Currently, ransomware is a serious threat challenging the computing environment, which needs instant attention to avoid moral and financial blackmail. Thus, there comes a real need for a novel technique that can identify and stop this kind of attack. Several earlier detection techniques followed a dynamic analysis method including a complex process. However, this analysis takes a long period of time for processing and analysis, during which the malicious payload is often sent. This study presents a new model of dwarf mongoose optimization with machine-learning-driven ransomware detection (DWOML-RWD). The presented DWOML-RWD model was mainly developed for the recognition and classification of goodware/ransomware. In the presented DWOML-RWD technique, the feature selection process is initially carried out using an enhanced krill herd optimization (EKHO) algorithm by the use of dynamic oppositional-based learning (QOBL). For ransomware detection, DWO with an extreme learning machine (ELM) classifier can be utilized. The design of the DWO algorithm aids in the optimal parameter selection of the ELM model. The experimental validation of the DWOML-RWD method can be examined on a benchmark dataset. The experimental results highlight the superiority of the DWOML-RWD model over other approaches.

Keywords: cybersecurity; artificial intelligence; internet of things; ransomware attack; dwarf mongoose optimization

1. Introduction

The internet of things (IoT) can be described as the increasing number of physical devices connected to the internet. Embedded with sensor nodes and software that can collect and share data online, IoT gadgets are invaluable in improving productivity and enhancing a massive number of processes throughout industries [1]. Inappropriately, the nature of IoT gadgets’ connectivity paves the way for malicious performers to take full advantage, especially because of the top ten vulnerabilities that make IoT gadgets...
insecure [2]. To achieve the scalability objective of the network, research workers in the past established several clustering methods. In a clustered network, every cluster can be assigned a gateway that sends packets to and from the base station (BS), considering that a cellular structure can be utilized for providing internet access to IoT nodes [3].

A node with good resources can be selected as a gateway. Compromising or capturing the gateway will affect every node in the cluster, and, thus, gateway security assaults should be protected against attacks [4]. Invaders usually encode victims’ data with their keys, and they decode the data or return the key only after the victims fulfill or pay their demands [5]. Previously, ransomware would simply disrupt the services and make the data inaccessible to the victim; until now, they were safely saved only on their system [6]. Now, data can also be stolen by attackers, and they threaten the victim to pay money. Additionally, it is not certain that the data can be retrieved when the ransom is paid or if the attackers leak the stolen data.

Ransomware is irreversible and tedious to stop, unlike other security issues. The approach of this malware depends on access limitations to user files via encryption and demands a ransom to attain the decryption key. An invader usually shows a ransom note after encoding the data of the victim, which generally indicates that the attack was executed; then, the invader demands money [7]. The attacker mentions the steps for making a payment with the ransom note that they offered. Ransomware attacks could cause trouble to the distributed IoT atmosphere and halt smooth work amongst heterogeneous data centers. Such mechanisms contain complicated structures of methods and corpora. Environments that are data centers have a huge magnitude of data and could pay money for avoiding damage to their reputation and exploitation of their data [8].

In the literature, three different studies are performed for ransomware detection: hybrid, static, and dynamic. Each analysis technique has pros and cons [9]. Dynamic analysis provides higher accuracy in detection through the execution of the samples. Deep learning (DL) and machine learning (ML) have effects on all aspects of life. Such technology has several applications in each domain because of its capability for decision making. Advanced assault and threat detection is easier and occurs in less time, because of the use of such effective technologies. DL is considered the optimal method for detecting the paradigms of an undergoing system [10].

ML and DL technologies have received considerable attention and are massively utilized in the advanced research of cybersecurity. ML technologies are utilized in the domain of ransomware detection, such as logistic regression (LR), random forest (RF), k-nearest neighbor (KNN), naïve Bayes (NB), and neural networks (NNs). In addition, DL models, such as deep neural networks (DNNs) and the deep autoencoder (AE), are also used for ransomware detection. Though several ransomware detection models are available in the literature, there is still a need to develop automated ransomware detection tools for enhanced performance. At the same time, the trial-and-error parameter-tuning process is difficult and erroneous. Therefore, a metaheuristics-based parameter-tuning process is essential.

This study presents a new model of dwarf mongoose optimization with machine-learning-driven ransomware detection (DWOML-RWD). The presented DWOML-RWD model was mainly developed for the recognition and classification of goodware/ransomware. In the presented DWOML-RWD technique, the feature selection process is initially carried out using an enhanced krill herd optimization (EKHO) algorithm by the use of dynamic oppositional-based learning (QOBL). For ransomware detection, DWO with an extreme learning machine (ELM) classifier was utilized. The design of the DWO algorithm aids in the optimal parameter selection of the ELM model. The experimental validation of the DWOML-RWD method can be examined on a benchmark dataset.

2. Related Works

In [11], the authors use DL methods to extract the latent representation of a high dimension of data, which is collected for identifying malicious behavior precisely. To be
specific, the method that the authors present depends on a hybrid-feature engineering method of variational and classical autoencoders. This hybrid method can be utilized for reducing the data dimension and extracts a good representation of the gathered system actions. Then, the new feature vector can be sent to a classifier that can be constructed based on batch-normalizing methods and a DNN. Aurangzeb et al. [12] develop BigRC-EML for ransomware classification and detection based on numerous dynamic and static features. The author leverage ensemble ML techniques on big data to predict the accuracy of ransomware detection. Though several ML techniques have been utilized in the identification of ransomware, the assessment of ensemble techniques has not yet been conducted. In addition, a novel feature-selecting technique related to PCA can be used to diminish the feature dimension.

Masum et al. [13] propose a feature-selection-related structure by adopting various ML methods, which include NN-related structures, to classify the security level for the prevention and detection of ransomware. The authors implement many ML techniques, namely, LR, DT, NB, NN-related classifiers, and RF, on a selected number of features for classifying ransomware. Ogundokun et al. [14] introduce an ML technique for detecting ransomware assaults on IoT gadgets. The study utilizes power for tracking and reviewing power utilization in 500 ms internals of every operating process. The paper utilizes three gadgets for performing the experiment: a projector, an android device, and a laptop computer. The method is used to monitor the power utilization of IoT gadgets utilizing several processes for the categorization of ransomware outside non-malicious functions.

The authors of [15] present a weighted minimum redundancy maximum relevance (WmRmR) method for the prediction of superior feature significance in data taken at the initial phases of ransomware assaults. The method integrates an improvised mRMR (EmRmR) with term frequency–inverse document frequency (TF-IDF); therefore, it can filter runtime and noisy conduct based on the weights computed by TF-IDF. Du et al. [16] demonstrate an intellectual KNN and density-related ML method for detecting ransomware pre-attacks on an endpoint mechanism. The feature engineering and data pre-processing methods are augmented with the KNN method to find the solution. The study of malware detection, utilizing advanced ML methods to advance highly proficient ransomware defensive solutions for the detection and prevention of ransomware pre-attack, aids endpoint security provider companies, anti-malware developers, vendors, and authors. Al-Hawawreh and Sitnikova [17] model a detection method related to stacked VAE and FC-NN, and they are able to study the latent structure of system actions and expose ransomware conduct. Moreover, the authors develop a data augmentation technique based on VAE to generate novel data that are employed in training an FC network for improvising the generalized abilities of the developed detection method.

3. The Proposed Model

In this article, a new DWOML-RWD method is introduced for cyberattack detection in the IoT environment. The presented DWOML-RWD model was mainly developed for the recognition and classification of goodware/ransomware. In the presented DWOML-RWD technique, the EKHO algorithm is used to perform the feature selection process. For ransomware detection, DWO with an ELM classifier is used. Figure 1 shows a block diagram of the DWOML-RWD approach.
3.1. Feature Selection: EKHO Algorithm

In the presented DWOML-RWD technique, the EKHO algorithm is used to perform the feature selection process. The KHO algorithm is inspired by the herding behavior of individual krill that is expressed by random diffusion, induced movement, and foraging activity [18]. For weight optimization, we employed the KHO technique; it is able to resolve optimization problems effectively. It is the more commonly employed heuristics optimization technique. This method imitates the krill individual (KI) behavior in KH. It contains two chief goals: (1) attaining food and (2) augmenting krill density. The position of KI can be affected by the following factors: (1) random diffusion; (2) movement inspired by other KI; and (3) foraging activity. Hence, the krill position is formulated using the Lagrangian model in the following:

\[
\frac{dU_i}{dt} = M_0 + F_a + D_f, \quad (1)
\]

where:
- \(M_0\) — movement guided by other KI.
- \(F_a\) — foraging movement.
- \(D_f\) — KI physical diffusion.

The steps are shown below.

Step 1: the motion of the direction of KI can be defined using repulsive swarm density, local swarm, and target swarm as follows:

\[
M^{\text{new}}_i = M^{m}_a \alpha_i + \omega_n M^{\text{old}}_i, \quad (2)
\]

where:
- \(M^{m}_a\) — maximum induced speed.
- \(\omega_n\) — motion inertia weight within \([0,1]\).
- \(M^{\text{old}}_i\) — last motion induced.

\(\alpha_i\) is estimated by Equation (3)

\[
\alpha_i = \alpha^{\text{local}}_i + \alpha^{\text{target}}_i \quad (3)
\]
\( \alpha_{local} \) denotes the local effects of a neighbor of \( i \)-th individuals, and \( \alpha_{target} \) represents the optimum solution direction from \( i \)th individuals.

\[
\alpha_{target}^i = I_b K_{ib} X_{ib}, \tag{4}
\]

In Equation (4), \( I_b \) indicates the coefficient and determined \( \alpha_{target} \) for \( i \)-th individuals.

\[
I_b = 2 \left( \frac{ra + 1}{l_{ma}} \right), \tag{5}
\]

where \( ra \) refers to the random number within \((0, 1)\).

Step 2: The foraging behavior can be estimated by

\[
F_{ai} = S_{pf} \gamma_i + \omega_f F_{ai}^{old}, \tag{6}
\]

where

\[
\gamma_i = \gamma_{i}^{best} + B_{i}^{best}, \tag{7}
\]

Here, \( S_{pf} \) denotes the foraging speed, \( \omega_f \) shows the inertia weight for foraging, and \( B_{i}^{best} \) represents the optimal solution.

Step 3: The physical diffusion of \( K_I \) is an arbitrary method, and the movement related to \( D_{fi} \) and \( \delta \) is evaluated in the following expression:

\[
D_{fi} = D_{fm} \delta, \tag{8}
\]

In Equation (8), \( D_{fi} \) represents the maximal diffusion speed, and \( \delta \) denotes the random directional vector and arrays of random numbers in \((-1, 1)\). The movement of the krill swarm determines the procedure that identifies the optimum fitness. Hence, the \( K_I \) location can be given as follows:

\[
U_j(t + \Delta t) = U_j(t) + \Delta t \frac{dU_j}{dt}. \tag{9}
\]

The variable \( \Delta t \) is crucial and regarded as a scaling factor of the speed vector. Hence, it should be modified regardless of the optimization problem. The value of \( \Delta t \) fully depends on the search space.

The EKHO algorithm was designed by using the DOBL concept. The OBL method was utilized for generating a unique opposition solution to the present solution [19]. It tries to produce the optimum solution, which leads to the speed rate of the convergences being enhanced. The opposite \((X^0)\) of a provided real number \((X \in [U, L])\) is measured as:

\[
X^0 = U + L - X \tag{10}
\]

The opposite point, \( X = [X_1, X_2, \ldots, X_{Dim}] \), is a point from \( Dim \)-dimensional searching space, \( X_1, X_2, \ldots, X_{Dim} \in R \), and \( X_j \in [U_j, L_j] \). Therefore, the opposite point \((X^0)\) of \( X \) is represented as:

\[
X^0_j = UB_j + L_j - X_j, \text{ where } j = 1 \ldots D. \tag{11}
\]

Compared with the opposite point, the dynamic opposite preference \((X^{DO})\) of value \( X \) is defined as:

\[
X^{DO}_j = X + w \times r_8 \left( r_9 \times X^0 - X \right), w > 0 \tag{12}
\]

where \( r_8 \) and \( r_9 \) signify the arbitrary values from the range of \([0,1]\), and \( w \) signifies the weighted agent. Therefore, the dynamic opposite value \((X^{DO})\) of \( X \) is equivalent to \([X_1, X_2, \ldots, X_{Dim}]\), which is written as:

\[
X^{DO}_j = X_j + w \times rand \left( rand \times X^0 - X_j \right), w > 0 \tag{13}
\]
Therefore, the DOBL optimizer starts by generating the primary solutions \(X = (X_1, \ldots, X_{\text{Dim}})\) and computes their dynamic opposite values \(X_{\text{Do}}\) utilized in Equation (13). Afterward, it is dependent upon the provided fitness value.

The fitness function (FF) employed in the EKHO technique contains a balance amongst the count of selective features from every solution (minimal) and classifier accuracy (maximum) achieved by using such selective features. Equation (14) defines the FF to estimate solutions.

\[
\text{Fitness} = \alpha \gamma_R(D) + \beta \frac{|R|}{|C|} \quad (14)
\]

in which \(\gamma_R(D)\) signifies the classifier error rate of a provided classification, \(|R|\) implies the cardinality of the chosen subset, and \(|C|\) indicates the entire count of features from a dataset; \(\alpha\), and \(\beta\) are two parameters corresponding to the significance of subset length and classifier quality. \(\in [1, 0]\) and \(\beta = 1 - \alpha\).

### 3.2. Ransomware Detection: ELM Model

For ransomware detection, the ELM classifier is used. In various fields, the ELM method is used for pattern recognition, wind speed forecasting, and fault diagnosis. The ELM method involves three layers: input, output, and hidden layers [20]. Figure 2 showcases the framework of ELM. By using the backpropagation model, the feedforward neural network must upgrade the weight in the iteration method; however, the ELM mechanism has a random initial weight that does not need to be upgraded in the iterative method. Consider that \(T = \{(c_i, e_i) | c_i \in \mathbb{R}^n, e_i \in \mathbb{R}^n\}\) denotes the training instances and \(F(.)\) indicates the activation function. The ELM method is formulated below:

\[
\begin{align*}
\sum_{j=1}^{N_h} \alpha_j F(\rho_j c_1 + \theta_j) &= e_1 \\
\sum_{j=1}^{N_h} \alpha_j F(\rho_j c_2 + \theta_j) &= e_2 \\
\sum_{j=1}^{N_c} \alpha_j F(\rho_j c_{N_c} + \theta_j) &= e_{N_c}
\end{align*}
\]  

Figure 2. Architecture of ELM.
Here, $\theta_i$ signifies the threshold of the hidden node; $\alpha_i$ signifies the weight linking the output and hidden nodes; $\rho_i$ shows the weight connecting the input and the hidden node; $c_i$ signifies the input sample of the module; $e_i$ denotes the output sample of models; and $N_s$ characterizes the sample count:

$$Y = \begin{bmatrix} F(\rho_1 c_1 + \theta_1) & \cdots & F(\rho_{N_h} c_1 + \theta_{N_h}) \\ \vdots & \ddots & \vdots \\ F(\rho_1 c_{N_s} + \theta_1) & \cdots & F(\rho_{N_h} c_{N_s} + \theta_{N_h}) \end{bmatrix}$$  \hspace{1cm} (17)

$$\alpha = [\alpha_1, \cdots, \alpha_{N_h}]^T$$  \hspace{1cm} (18)

$$e = [e_1, \ldots, e_{N_s}]^T$$  \hspace{1cm} (19)

Here, $e$ denotes the sample label matrix; $Y$ indicates the hidden layer output matrixes; $\alpha$ indicates the weight matrix; and $N_h$ shows the node count in a hidden state. The hidden layer threshold and input weight of the ELM algorithm are determined randomly. In the iteration model, the weight vector does not need to be modified; only the output weight needs to be evaluated and is given below:

$$\alpha = (Y^T Y)^{-1} Y^T e$$  \hspace{1cm} (20)

The ELM predictive method can be attained.

$$e = \sum_{i=1}^{N_h} \alpha_j F(\rho_i c + \theta_i)$$  \hspace{1cm} (21)

The computation method of ELM is given as follows:

1. Define the model sample;
2. Initialize the hidden layer threshold and input weight randomly;
3. Compute the output matrix;
4. Resolve the output weight of the hidden layer.

### 3.3. Parameter Optimization: DWO Algorithm

In this work, the design of the DWO algorithm aids in the optimal parameter selection of the ELM model. This model mimics the behavior of the dwarf mongoose while searching for its food. Usually, DWO initiates by fixing the first value in a sequence of solutions through the subsequent Equation [21]:

$$x_{i,j} = l_j + \text{rand} \times (u_j - l_j)$$  \hspace{1cm} (22)

In Equation (22), $\text{rand}$ denotes an arbitrary value integrated within $[0,1]$, while $u_j$ and $l_j$ indicate the limit of the search domain. The swarming of DWO comprises three groups, namely, babysitters, the alpha group, and scouts. Every group has its own means of capturing food, and this is discussed below:

- **Alpha Group**

  The fitness of every solution can be calculated when the population is introduced. Equation (23) computes the probability value for the fitness of the population, and ($\alpha$) the alpha female is selected according to the probability

$$\alpha = \frac{\text{fit}_i}{\sum_{i=1}^{n} \text{fit}_i}$$  \hspace{1cm} (23)

$n$ corresponds to the mongoose counts in the alpha group. The babysitter’s number is represented as bs. Peep shows the vocalization of the leading female, which keeps the family on track.
Every mongoose sleeps in the initial sleeping mounds that are fixed $\emptyset$. The DWO applies the expression for generating a candidate food location.

$$X_{i+1} = X_i + \phi \times \text{peep}$$

(24)

The sleeping mound is given as follows in every reiteration, in which $\phi$ indicates a uniform distribution random number $[-1, 1]$.

$$sm_i = \frac{fit_{i+1} - fit_i}{\max \{||fit_{i+1}, fit_i||\}}$$

(25)

Equation (26) encompasses the average value of the sleeping mound.

$$\varphi = \frac{\sum_{i=1}^{n} sm_i}{n}$$

(26)

When the babysitting exchange condition is satisfied, it progresses to the scouting stage; wherever the sleeping mound or following food source is, is taken into account.

• Scouts

Since mongooses are known to not return to earlier sleeping mounds, the scout looks for the sleeping mounds, which ensures exploration. For this method, foraging and scouting are implemented simultaneously. This motion is modeled after an unsuccessful or successful search for sleeping mounds. In other words, the movement of mongooses is dependent on the overall accuracy. In that regard, if the family forages some distance, they come to a novel sleeping mound as shown below.

$$X_{i+1} = \begin{cases} 
X_i - CF \times \phi \times \text{rand} \times \left[ X_i - \bar{M} \right] & \text{if } \varphi_{i+1} > \varphi_i \\
X_i + CF \times \phi \times \text{rand} \times \left[ X_i - \bar{M} \right] & \text{else}
\end{cases}$$

(27)

In Equation (27), rand denotes an arbitrary value within $[0, 1]$, and $CF = \left(1 - \frac{\text{iter}}{\text{maxiter}}\right)^2 \left(1 - \frac{\text{iter}}{\text{maxiter}}\right)$, where the variable that regulates the mongoose group collective-volitive motion is linearly reduced as the iteration progresses. $\bar{M} = \Sigma_{i=1}^{n} X_i \times sm_i / X_i$, whereby the mongoose motion to the novel sleeping mounds is defined.

• Babysitters

Usually, babysitters are small group members that remain with the young and are cycled daily to assist the alpha female (mother) in leading the remaining group on regular foraging expeditions. The fitness weight is fixed as zero, which ensures that the average weight of the alpha group is reduced in the following iteration, which obstructs the movement of the group and intensifies exploitation. Algorithm Al encompasses the pseudo-code for the proposed method.

4. Results and Discussion

The proposed model was simulated using a Python 3.6.5 tool on PC i5-8600k, GeForce 1050Ti 4GB, 16GB RAM, 250GB SSD, and 1TB HDD. The parameter settings are as follows: learning rate, 0.01; dropout, 0.5; batch size, 5; epoch count, 50; and activation, ReLU. The experimental assessment of the DWOML-RWD algorithm was carried out utilizing a dataset comprising 840 samples. The dataset holds 420 goodware samples and 420 ransomware samples as depicted in Table 1.
Table 1. Dataset details.

<table>
<thead>
<tr>
<th>Class</th>
<th>No. of Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goodware</td>
<td>420</td>
</tr>
<tr>
<td>Ransomware</td>
<td>420</td>
</tr>
<tr>
<td><strong>Total Number of Samples</strong></td>
<td><strong>840</strong></td>
</tr>
</tbody>
</table>

Figure 3 illustrates the set of confusion matrices formed by the DWOML-RWD model in different runs. With run-1, the DWOML-RWD model classified 412 samples into goodware and 417 samples into ransomware. At the same time, with run-2, the DWOML-RWD model classified 413 samples into goodware and 416 samples into ransomware. Simultaneously, with run-3, the DWOML-RWD model classified 415 samples into goodware and 418 samples into ransomware. Additionally, with run-4, the DWOML—RWD model classified 414 samples into goodware and 418 samples into ransomware. Lastly, with run-5, the DWOML-RWD model classified 416 samples into goodware and 419 samples into ransomware.

Figure 3. Confusion matrices of DWOML-RWD approach: (a) run-1, (b) run-2, (c) run-3, (d) run-4, and (e) run-5.
Table 2 displays the overall classification results of the DWOML-RWD model on ransomware detection. Figure 4 reports a brief ransomware classification performance of the DWOML-RWD model in terms of accuracy, sensitivity, specificity, and F-score. The results imply that the DWOML-RWD model ensured enhanced performance under both classes. For instance, on run-1, the DWOML-RWD model attained an average accuracy, sensitivity, specificity, and F-score of 98.69%, 98.69%, 98.69%, and 98.69%, respectively. Meanwhile, on run-2, the DWOML-RWD model gained an average accuracy, sensitivity, specificity, and F-score of 98.69%, 98.69%, 98.69%, and 98.69%, respectively. Furthermore, on run-3, the DWOML-RWD model achieved an average accuracy, sensitivity, specificity, and F-score of 99.17%, 99.17%, 99.17%, and 99.17%, respectively. Further, on run-4, the DWOML-RWD model acquired an average accuracy, sensitivity, specificity, and F-score of 99.05%, 99.05%, 99.05%, and 99.05%, respectively. Lastly, on run-5, the DWOML-RWD model obtained an average accuracy, sensitivity, specificity, and F-score of 99.40%, 99.40%, 99.40%, and 99.40%, respectively.

Table 2. Result analysis of DWOML-RWD technique with distinct measures and runs.

<table>
<thead>
<tr>
<th>Class Labels</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>F-Score</th>
<th>FNR</th>
<th>FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Goodware</td>
<td>98.69</td>
<td>98.10</td>
<td>99.29</td>
<td>98.68</td>
<td>01.90</td>
<td>00.71</td>
</tr>
<tr>
<td>Ransomware</td>
<td>98.69</td>
<td>99.29</td>
<td>98.10</td>
<td>98.70</td>
<td>00.71</td>
<td>01.90</td>
</tr>
<tr>
<td>Average</td>
<td>98.69</td>
<td>98.69</td>
<td>98.69</td>
<td>98.69</td>
<td>01.31</td>
<td>01.31</td>
</tr>
<tr>
<td>Run-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Goodware</td>
<td>98.69</td>
<td>98.33</td>
<td>99.05</td>
<td>98.69</td>
<td>01.67</td>
<td>00.95</td>
</tr>
<tr>
<td>Ransomware</td>
<td>98.69</td>
<td>99.05</td>
<td>98.33</td>
<td>98.70</td>
<td>00.95</td>
<td>01.67</td>
</tr>
<tr>
<td>Average</td>
<td>98.69</td>
<td>98.69</td>
<td>98.69</td>
<td>98.69</td>
<td>01.31</td>
<td>01.31</td>
</tr>
<tr>
<td>Run-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Goodware</td>
<td>99.17</td>
<td>98.81</td>
<td>99.52</td>
<td>99.16</td>
<td>01.19</td>
<td>00.48</td>
</tr>
<tr>
<td>Ransomware</td>
<td>99.17</td>
<td>99.52</td>
<td>98.81</td>
<td>99.17</td>
<td>00.48</td>
<td>01.19</td>
</tr>
<tr>
<td>Average</td>
<td>99.17</td>
<td>99.17</td>
<td>99.17</td>
<td>99.17</td>
<td>00.83</td>
<td>00.83</td>
</tr>
<tr>
<td>Run-4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Goodware</td>
<td>99.05</td>
<td>98.57</td>
<td>99.52</td>
<td>99.04</td>
<td>01.43</td>
<td>00.48</td>
</tr>
<tr>
<td>Ransomware</td>
<td>99.05</td>
<td>99.52</td>
<td>98.57</td>
<td>99.05</td>
<td>00.48</td>
<td>01.43</td>
</tr>
<tr>
<td>Average</td>
<td>99.05</td>
<td>99.05</td>
<td>99.05</td>
<td>99.05</td>
<td>00.95</td>
<td>00.95</td>
</tr>
<tr>
<td>Run-5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Goodware</td>
<td>99.40</td>
<td>99.05</td>
<td>99.76</td>
<td>99.40</td>
<td>00.95</td>
<td>00.24</td>
</tr>
<tr>
<td>Ransomware</td>
<td>99.40</td>
<td>99.76</td>
<td>99.05</td>
<td>99.41</td>
<td>00.24</td>
<td>00.95</td>
</tr>
<tr>
<td>Average</td>
<td>99.40</td>
<td>99.40</td>
<td>99.40</td>
<td>99.40</td>
<td>00.60</td>
<td>00.60</td>
</tr>
</tbody>
</table>

Figure 5 demonstrates the detailed ransomware classification performance of the DWOML-RWD approach in terms of FNR and FPR. The results denote that the DWOML-RWD approach ensured enhanced performance under both classes. For example, on run-1, the DWOML-RWD model attained an average FNR and FPR of 1.31% and 1.31%, respectively. In parallel, on run-2, the DWOML-RWD model obtained an average FNR and FPR of 1.31% and 1.31%, respectively. Additionally, on run-3, the DWOML-RWD model gained an average FNR and FPR of 0.83% and 0.83%, respectively. Further, on run-4, the DWOML-RWD model reached an average FNR and FPR of 0.95% and 0.95%, respectively. Lastly, on run-5, the DWOML-RWD model obtained an average FNR and FPR of 0.60% and 0.60%, respectively.
Figure 4. Average analysis of DWOML-RWD approach with distinct measures.

Figure 5. Average FNR and FPR analysis of DWOML-RWD approach with distinct runs.

The training accuracy (TRA) and validation accuracy (VLA) acquired by using the DWOML-RWD approach on the test dataset are exemplified in Figure 6. The experimental outcome represented by the DWOML-RWD technique attained higher values of TRA and VLA. Seemingly, the VLA is greater than TRA.
Figure 5. Average FNR and FPR analysis of DWOML-RWD approach with distinct runs.

Figure 6. TRA and VLA analysis of DWOML-RWD approach.

The training loss (TRL) and validation loss (VLL) attained by using the DWOML-RWD method on the test dataset are exhibited in Figure 7. The experimental outcome denotes the DWOML-RWD approach exhibited the lowest values of TRL and VLL. Particularly, the VLL is lower than the TRL.

Figure 7. TRL and VLL analysis of DWOML-RWD approach.
A clear precision–recall inspection of the DWOML-RWD approach on the test dataset is portrayed in Figure 8. The figure exemplifies that the DWOML-RWD methodology resulted in enhanced values of precision–recall values under all classes.

Figure 8. Precision–recall analysis of DWOML-RWD approach.

A brief ROC analysis of the DWOML-RWD approach on the test dataset is portrayed in Figure 9. The results imply that the DWOML-RWD method displayed an ability to categorize distinct classes on the test dataset.

Figure 9. ROC curve analysis of DWOML-RWD approach.
Table 3 presents the ransomware detection results of the DWOML-RWD model compared with other ML models. These results indicate that the DWOML-RWD model displays better performance than the other methods [22]. Figure 10 provides a comparative sensitivity and specificity inspection of the DWOML-RWD technique with other existing approaches. The figure reveals that the DWOML-RWD model showed improved performance in all aspects. For instance, based on sensitivity, the DWOML-RWD approach gained enhanced sensitivity of 99.40%, whereas the Adaboost-M1, bagging, ROF, RF, and DT methodologies resulted in lower sensitivity values of 94.60%, 93.80%, 96.90%, 99.21%, and 98.12%, respectively.

Table 3. Comparative analysis of DWOML-RWD technique with existing algorithms.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>FNR</th>
<th>FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DWOML-RWD</td>
<td>99.40</td>
<td>99.40</td>
<td>99.40</td>
<td>0.60</td>
<td>0.60</td>
</tr>
<tr>
<td>AdaBoost-M1</td>
<td>96.13</td>
<td>94.60</td>
<td>95.00</td>
<td>5.40</td>
<td>5.00</td>
</tr>
<tr>
<td>Bagging</td>
<td>99.00</td>
<td>93.80</td>
<td>96.50</td>
<td>6.20</td>
<td>3.50</td>
</tr>
<tr>
<td>Rotation Forest</td>
<td>96.24</td>
<td>96.90</td>
<td>97.40</td>
<td>3.10</td>
<td>2.60</td>
</tr>
<tr>
<td>Random Forest</td>
<td>98.81</td>
<td>99.21</td>
<td>98.70</td>
<td>0.79</td>
<td>1.30</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>97.68</td>
<td>98.12</td>
<td>98.51</td>
<td>1.88</td>
<td>1.49</td>
</tr>
</tbody>
</table>

Figure 10. Sensitivity and specificity analysis of DWOML-RWD approach with existing algorithms. Based on specificity, the DWOML-RWD method gained enhanced specificity of 99.40%, whereas the Adaboost-M1, bagging, ROF, RF, and DT algorithms resulted in lower specificity values of 95%, 96.50%, 97.40%, 98.70%, and 98.51%, respectively.

Figure 11 provides a comparative FNR and FPR inspection of the DWOML-RWD algorithm with other existing approaches. The figure reveals that the DWOML-RWD model shows improved performance in all aspects. For instance, based on FNR, the DWOML-RWD approach attained a minimal FNR of 0.60%, whereas the Adaboost-M1, bagging, ROF, RF, and DT models resulted in increased FNR values of 5.40%, 6.20%, 3.10%, 0.79%, and 1.88%, respectively. Additionally, based on PNR, the DWOML-RWD approach gained a minimal PNR of 0.60%, whereas the Adaboost-M1, bagging, ROF, RF, and DT techniques resulted in
increased PNR values of 5%, 3.50%, 2.60%, 1.30%, and 1.49, respectively. From these results, it was confirmed that the DWOML-RWD model assured enhanced performance.

Figure 10. Sensitivity and Specificity analysis of DWOML-RWD approach with existing algorithms. Based on Specificity, the DWOML-RWD method gained enhanced Specificity of 99.40%, whereas the Adaboost-M1, bagging, ROF, RF, and DT algorithms resulted in lower Specificity values of 95%, 96.50%, 97.40%, 98.70%, and 98.51%, respectively.

Figure 11 provides a comparative FNR and FPR inspection of the DWOML-RWD algorithm with other existing approaches. The figure reveals that the DWOML-RWD model shows improved performance in all aspects. For instance, based on FNR, the DWOML-RWD approach attained a minimal FNR of 0.60%, whereas the Adaboost-M1, bagging, ROF, RF, and DT models resulted in increased FNR values of 5.40%, 6.20%, 3.10%, 0.79%, and 1.88, respectively. Additionally, based on PNR, the DWOML-RWD approach gained a minimal PNR of 0.60%, whereas the Adaboost-M1, bagging, ROF, RF, and DT techniques resulted in increased PNR values of 5%, 3.50%, 2.60%, 1.30%, and 1.49, respectively. From these results, it was confirmed that the DWOML-RWD model assured enhanced performance.

5. Conclusions

In this article, a new DWOML-RWD method was introduced for cyberattack detection in the IoT environment. The presented DWOML-RWD model was mainly developed for the recognition and classification of goodware/ransomware. In the presented DWOML-RWD technique, the EKHO algorithm was used to perform the feature selection process. For ransomware detection, DWO with an ELM classifier was used. The design of the DWO algorithm aids in the optimal parameter selection of the ELM model. The experimental validation of the DWOML-RWD method can be examined on a benchmark dataset. The experimental outcomes highlight the superiority of the DWOML-RWD model over other approaches. Thus, the DWOML-RWD approach can be utilized for ransomware detection in a real-time IoT-cloud environment. In the future, the DWOML-RWD model can be extended to the integration of hybrid deep learning models.

Author Contributions: Conceptualization, K.A.A. and D.H.E.; methodology, K.T.; software, A.Y.; validation, R.A., O.A. and A.M.; formal analysis, A.Y.; investigation, K.T.; resources, M.A.D.; data curation, O.A.; writing—original draft preparation, K.A.A., D.H.E. and K.T.; writing—review and editing, A.M., A.Y., O.A.; visualization, R.A.; supervision, K.A.A.; project administration, M.A.D.; funding acquisition, D.H.E. All authors have read and agreed to the published version of the manuscript.

Funding: Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R238), 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 (Grant Code: 22UQU4331004DSR03).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing does not apply to this article as no datasets were generated during the current study.
Conflicts of Interest: The authors declare that they have no conflict of interest.

References


22. Khammas, B.M. Ransomware detection using random forest technique. *ICT Express* 2020, 6, 325–331. [CrossRef]