International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V71I10P216

Volume 71 Issue 10, 173-182, October 2023

Original Article

Binary Mayfly Optimization with Deep Wavelet Network-Based Malware Detection for Cybersecurity

V. S. Pavankumar[1], S. Arivalagan[2], M. Murugesan[3], P. Sudhakar[4]

[1,2,4]*Department of CSE, Annamalai University, Chidambaram, India.*
[3]*Department of CSE, Anurag Engineering College, Telangana, India.*

[1]*Corresponding Author : pavankumar620@gmail.com*

Received: 24 June 2023 Revised: 28 August 2023 Accepted: 15 September 2023 Published: 03 October 2023

Abstract *- A malware attack is the most prominent cyberattack where malware (malicious software) implements unauthorized action on the target's system. The malware (otherwise called a virus) incorporates different attacks like spyware, command and control, ransomware, etc. Cyber attackers create, sell and use malware for various reasons; however, it is more commonly used to steal business, personal or financial data. Machine Learning (ML) approaches, and Deep Learning (DL) approaches are currently utilized to give an effective solution to overcome these cyberattacks. With the advancement of the ML and DL approaches, a classification model has been commonly exploited in this study to categorize whether the file is malicious or not. This article introduces a new Binary Mayfly Optimization with Deep Wavelet Network-based Malware Detection (BMFO-DWNMD) for cybersecurity. The presented BMFO-DWNMD technique focuses on the recognition and classification of malware using the classification and Feature Selection (FS) process. In the proposed BMFO-DWNMD approach, the BMFO approach is exploited for the optimum Feature Subset (FSB) selection. Next, the BMFO-DWNMD model uses a DWN classifier to recognize malware attacks. Lastly, the African Vulture Optimization Algorithm (AVOA) is exploited for the process of hyperparameter tuning. A comprehensive set of simulations has been performed to depict the investigational validation of the BMFO-DWNMD model. The experimental outcomes illustrate an enhanced achievement of the BMFO-DWNMD model over other models.*

Keywords *- Cybersecurity, Malware attacks, Mayfly optimization, Feature selection, Deep learning.*

1. Introduction

Cyber security experts focused more on risk evaluation and devised methods for mitigating. Devising a successful approach was a plan set in the cyber defence area. ML even turned out to be a significant concern in data protection through ML, which was successful in cyber defence [1]. The exponential growth in Cloud Computing (CC), evolutionary computation and networking resulted from unprecedented advances in computational, computing, and storage technology [2]. As the earth is being digitalized rapidly - there occurs an increasing need for comprehensive and sophisticated data security and privacy problems to fight security threats, which becomes more complex [3].

Today, huge volumes of data are produced all over the grids, raising accessibility for real-time system monitoring. Sightseeing such data significantly boosts the prognosis, performance monitoring, and diagnosis of anomalies in complicated systems [4]. Historical data describing the system's operation can assist in finding anomalies and effective attacks. But, conventional Bad Data Detection (BDD) methods are unprepared for real-time storage and computational problems due to the massive quantity of data produced in the smart grid [5].

Such difficulties uncover the possibility of utilizing data analytical approaches, like ML, for managing complicated structure data sets with AI to prevent and detect cyber-attacks [6]. ML techniques are used for analyzing different combinations of measurements using control actions, AMI, and states by learning their paradigms [7]. It can identify False Data Injection (FDI) assault by learning the non-linear, complicated relations between measurements. As mentioned in the studies, this can be made similar to successful methods implemented for other power system problems. It is known to predict software change proneness, categorize network traffic, DDoS detection, perform software metrics estimation, Botnet detection, etc. [8].

The success of DL in big data areas can be implemented to fight cyber threats as mutations of assaults are like minor changes in, for example, imagery pixels [9]. It denotes that DL in security learns the true face (legitimate or attack) of cyber data on even minor changes, representing the resiliency of DL

to minor changes in network data by constituting a high-level invariant representation of the training dataset [10].

This study designs a new Binary Mayfly Optimization with Deep Wavelet Network-based Malware Detection (BMFO-DWNMD) for cybersecurity. The presented BMFO-DWNMD technique focuses on the recognition and classification of malware using the classification and Feature Selection (FS) process. In the proposed BMFO-DWNMD approach, the BMFO approach is exploited for the optimum FSB selection. Next, the BMFO-DWNMD model uses a DWN classifier to recognize malware attacks. Lastly, the African Vulture Optimization Algorithm (AVOA) is exploited for the process of hyperparameter tuning. A comprehensive set of simulations has been performed to depict the investigational validation of the BMFO-DWNMD model.

2. Literature Review

Al-Abassi et al. [11] present a DL technique for constructing novel balanced representations of imbalanced databases. A novel representation is provided as an ensemble DL attack detection method specially planned for the ICS platform. The presented attack detection scheme leverages Decision Tree (DT) and Deep Neural Networks (DNNs) techniques for identifying cyber-attacks from novel representations.

Sarker et al. [12] presented an Intrusion Detection Tree ("IntruDTree") ML-oriented safety method that initially considered the hierarchy of privacy aspects as per their significance and then framed a tree-oriented standardized ID method related to the selective significant attributes. This technique was ineffective concerning the prediction outcome for hidden test cases but even lessened the computing difficulty of the method by diminishing the feature dimension.

In [13], modelled a data-driven cyberattack recognition technique for islanded DC microgrids. Data can be gathered by observing the performance of intellectual attackers who can bypass the classical cyberattack recognition systems and disturb the system's operation. The Reinforcement Learning (RL) method verifies these intellectual attackers' activities, who use the vulnerability of index-related cyberattack recognition algorithms, like discordant detection systems.

In [14], the author provides the complete expansion of a novel intellectual and independent DL-related detection and classifier mechanism for cybercrime in IoT communication networking that uses the supremacy of CNN, IoT-related ID and Classifier System utilizing CNN (IoT-IDCS-CNN). The presented IoT-IDCS-CNN achieves calculation that uses the vigorous Compute Unified Device Architecture (CUDA) based NVIDIA GPU and parallel processing that uses I9-core-related high-speed Intel CPUs.

Elsisi et al. [15] presented a powerful IoT structure for monitoring the online status of Gas-Insulated Switchgear (GIS) rather than the conventional observing techniques. The presented IoT structure can be extracted from the Cyber-Physic System, namely CPS in Industry 4.0 conception. However, the categorization of the cyberattacks and GIS insulation shortcomings indicates core difficulties against the application of IoTs for tracking the GIS status and online monitoring. So, advanced ML approaches were used to identify cyber-attacks to verify and paradigm. In [16, 17], the cognitive ML-assisted Attack Detection structure was modelled for securely sharing medical datasets. The Healthcare CPS will effectively spread the gathered data to cloud storage. This presented method depends on the patient-centric model that protects the data on trust-worthy devices such as end-user control data sharing access and end-users' mobile phones.

3. The Proposed Model

The present article proposes an innovative BMFO-DWNMD methodology for automatic malware recognition to accomplish cybersecurity. The presented BMFO-DWNMD methodology concentrated on classifying and detecting malware using FS and the classifying process. Fig. 1 demonstrates the comprehensive procedure of the BMFO-DWNMD algorithm.

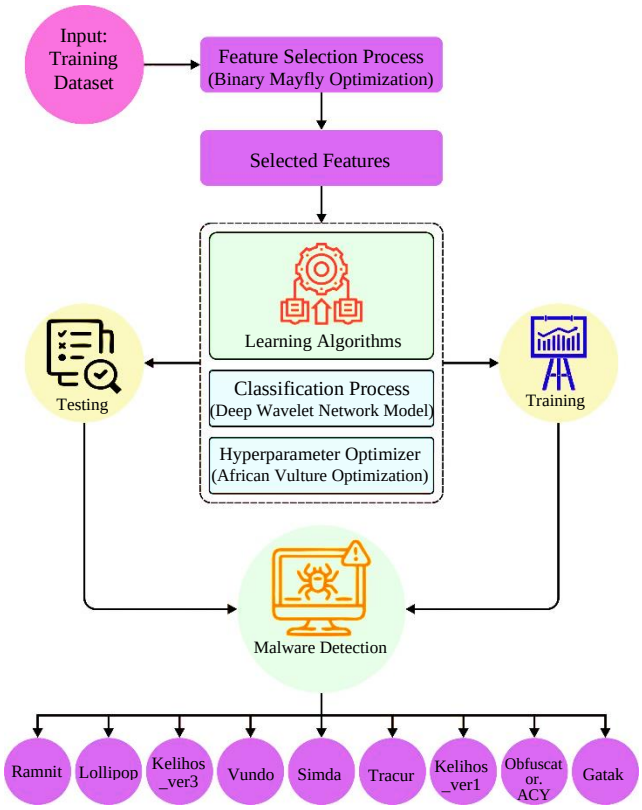

Fig. 1 Overall procedure of BMFO-DWNMD system

3.1. Module I: MFOFS Technique

In the presented BMFO-DWNMD technique, the BMFO technique is exploited for the optimum FSB selection. The MFO was established to optimize the non-interrupted search space [18]. Though, the nature of FS problems is based on an initial binary solution in the feature space.

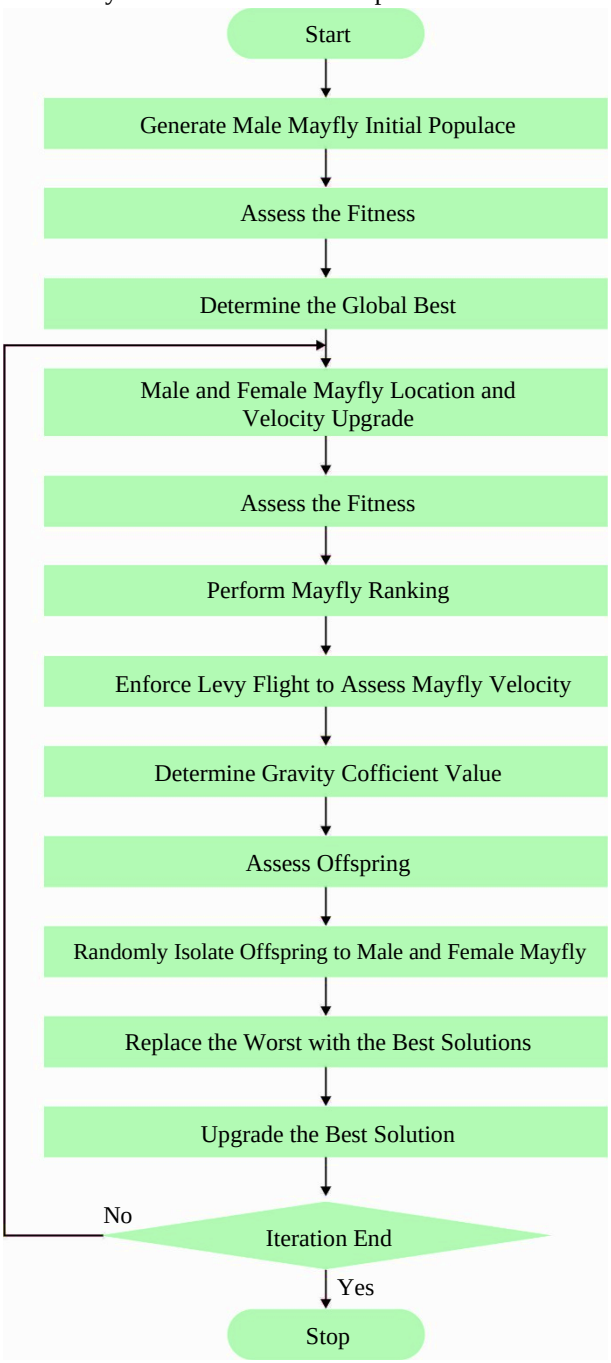

Fig. 2 MFO flow

As a result, certain operators of the MFO must be adapted to let the model for optimizing in the binary searching range. Calculating the length amid global and current solutions is the most popular swarming behaviour (for example, PSO). Based on this, it has been demonstrated that the transfer function is the simplest operator that is used for converting the continuous optimizer into binary. They are applied without any modifications to the essence and structure of the model. The transfer function maps the updating process to a binary updating process. Accordingly, the component of the relocated solution is a constraint to the "zero" or "one". In this work, the sigmoidal transfer function is exploited to convert the typical MFO as Binary MFO represents the (BMFO).

The sigmoid function's primary goal is to describe a possibility for all the components of the solutions ranging from zero to one. Eberhart and Kennedy used to change the PSO into the binary search. All the moths will upgrade their location in the binary feature space by Eq. (3) according to the probabilities attained from Eq. (2). The binary version of BMFO has been demonstrated in Algorithm 1.

$$\Delta M = D_i \times e^{bt} \times cos(2\Pi) \tag{1}$$

$$T(\Delta M_t) = \frac{1}{1 + e^{\Delta M_t}} \tag{2}$$

$$M_i^d(t+1) = \begin{cases} 0, if\ rand < TF(M) \\ 1, if\ rand \geq TF(M) \end{cases} \tag{3}$$

Algorithm 1: Pseudocode of BMFO

Input: Max_iteration, n and d represents the moth and dimension numbers
Output: Estimated global solution
Initializing the location of the moth
While $l \leq Max_$iteration do
Upgrade flame number
$OM = FitnessFunction(M)$;
if $l == 1$ then
$F =$ sort (M);
$OF =$ sort (OM);
Else
$F =$ sort (M_{l-1}, M_l);
$OF =$ sort (OM_{l-1}, OM_l);
End if
For $i = 1$: n do
For $j = 1$: d do
Upgrade r and t;
Evaluate D in terms of the respective moth;
Upgrade the step vector of moth ΔM.
Evaluate the probability.
Upgrade the location vector of the moth
End for
End for
$l = l + 1$;
End while

In the study, the classifier of KNN is utilized for evaluating every feature set. The objective of FS is to instantaneously accomplish two contrary targets, which are minimizing and maximizing the feature numbers and accuracy of the classification. Now, the Fitness Function (FF) in Eq. (4) is exploited for evaluating the carefully chosen subset in every approach where $\alpha\gamma_R(D)$ denotes the error rate, $|R|$ indicates the chosen feature numbers, $|C|$ shows the feature numbers in the original data, and $\alpha \in [0,1]$, $\beta = (1-\alpha)$ represents two variables that provide an indication regarding the length of FSB and importance of classification based on the recommendation. Fig. 2 illustrates the flowchart of MFO.

$$Fitness = \alpha\gamma_R(D) + \beta\frac{|R|}{|C|} \tag{4}$$

3.2. Module II: DWN-based Classification Model

At this stage, the BMFO-DWNMD technique uses a DWN classifier to recognize malware attacks. A Wavelet network is the fusion between neural and wavelet networks [19]. It is composed of three layers: an initial layer with N_i entries, a Hidden Layer (HL) comprising N_w wavelets and the output layer that accommodates the weighted output of the wavelet networks. It exploits a feedforward propagation technique from input to output neurons.

Moreover, it contains a specific vicinity with the structure of the NN. The major comparison between both networks is that they compute linear incorporation of non-linear functions relying on these combinations' adjustable parameters (dilation and translation). But, the transfer function exploited by the hidden cells is their main difference. The steps for constructing the Deep Wavelet Network (DWN) are:

Step 1: Generate a wavelet network using a single HL where the transfer function depends on the wavelet families.
Step 2: Generate an additional wavelet network with the last layer detached to create the features attained in the initial HL.
Step 3: Perform training for the second AE through the feature produced from the initial AE (Step 2).
Step 4: Eliminate the last layer for generating the features attained in the second HL.
Step 5: Stack the encoder from the AE together to form a DNN.
Step 6: Steps 3, 4, and 5 are repeated based on the desired amount of HLs.

Afterwards, in the learning stage, organize the vector attained in the matrix where every column signifies a picture such that the classification stage can be employed:

Where W_{ij} indicates the weight connecting the i^{th} and j^{th} output of neurons. a_i shows the dilation coefficient of the i^{th} neuron, bi denotes the translation coefficient of the i^{th} neuron. The method depends on two HLs whereby the wavelet basis function operates as the activation function ψ. During the learning stage, the feedforward propagation algorithm is applied. The usage of the model is to decrease the error generated by the network, thereby correcting this parameter that was determined previously: the translations, weights and dilations. Then, applied the quadratic cost function for measuring these errors:

$$E = \frac{1}{2}\sum_{t=1}^{T}(y_d(t) - y(t))^2 \tag{5}$$

In Eq. (5), $y(t)$ represents the output provided by the network; $y_d(t)$ indicates the preferred output, and it can be formulated by:

$$y(t) = \sum_{s=1}^{S} w_s \cdot \psi_s\left(\frac{t-b_s}{a_s}\right) \tag{6}$$

The GD algorithm is selected for the error minimization technique. In all the iterations, the images are proposed to the input or output network and later transmit the computation of the layer to others until the layer of output. It comprises altering the setting in the reverse direction to the error gradient function:

$$V_{t+1} = V_t - \varepsilon(t)\frac{\partial E}{\partial V} \tag{7}$$

In Eq. (7) V_t and $\varepsilon(t)$ signifies the parameter $\{w, a, b\}$ and pitch of gradient at the t^{th} iteration. While placing $e(t) = y_d(t) - y(t)$, the following function is derived:

$$\frac{\partial E}{\partial \omega_{ij}} = \sum_{t=1}^{T} e\,(t)\psi(\tau) \tag{8}$$

$$\frac{\partial E}{\partial a_i} = \sum_{t=1}^{T} e\,(t)\omega_{ij}\frac{\partial \psi(\tau)}{\partial a_i} \tag{9}$$

$$\frac{\partial E}{\partial b_i} = \sum_{t=1}^{T} e\,(t)\omega_{ij}\frac{\partial \psi(\tau)}{\partial b_i} \tag{10}$$

Where $\tau = \frac{t-b}{a}$. The parameter $\{w, a, b\}$ has been randomly initialized. The alteration of this setting is accomplished as follows:

$$\omega(t+1) = \omega(t) + \mu_w \triangle \omega \tag{11}$$

$$a(t+1) = a(t) + \mu_a \triangle a \tag{12}$$

$$b(t+1) = b(t) + \mu_b \triangle b \tag{13}$$

From the expression, μ_w, μ_a and μ_b denotes the rate of learning rate of the three networking setups.

$$\triangle \omega = -\frac{\partial E}{\partial \omega}, \triangle b = -\frac{\partial E}{\partial b}, \triangle a = -\frac{\partial E}{\partial a}$$

3.3. Module III: AVOA-Based Parameter Optimization

Finally, the AVOA is exploited for the hyperparameter tuning process. The AVOA is a metaheuristic technique based on the navigational and feeding behaviours of African vultures [20]. Also, the presented algorithm is extremely adaptable and flexible when compared to other metaheuristic approaches and has a lower computation difficulty. The exploitation and exploration phases of AVOA are explained briefly in the following. The possibility of selecting a vulture for guiding others to one of the optimum solutions in each ensemble is evaluated as follows:

$$P(i+1) = \begin{cases} R(i) - |X \times R(i) - P(i)| \times F & if P_1 \geq rand_{p_1} \\ R(i) - F + r_1 \times \big((U-L) \times r_2 + L\big) & if\ P_1 < rand_{p_1} \end{cases} \quad (14)$$

In Eq. (14), $P(i)$ indicates the location of the vulture in the existing iterations and $P(i+1)$ denotes the location at the following iterations, correspondingly. Furthermore, F denotes the satiation vulture rate, U and L indicate the upper and lower bounds, whereas r_1 and r_2 denote the random variable, and X shows the vector that depicts the random movement of the vulture. Furthermore, $rand_{p_1}$ is a randomly generated value within [0, 1]:

$$R(i) = \begin{cases} Best\ Vulture_1 & if\ P_i = L_1 \\ Best\ Vulture_2 & if\ P_i = L_2 \end{cases} \quad (15)$$

In Eq. (15), Best $Vulture_1$ and Best $Vulture_2$ denote the first and second-group optimum solution at the existing iteration. Beforehand the optimization search starts, both L_1 and L_2 are initialized where the value lies within [0, 1] and the sum equals 1. Two approaches are provided in this stage based on the satiation vulture rate. Once the vulture (F) 's satiation rate is superior or equivalent to 0.5, then the vulture fights for food in the circular motion that is evaluated as follows:

$$P(i+1) = \begin{cases} |X \times R(i) - P(i)| \times (F + r_3) - \big(R(i) - P(i)\big) & if\ P_2 \geq rand_{p_2} \\ R(i) - (S_1 + S_2) & if\ P_2 < rand_{p_2} \end{cases} \quad (16)$$

Where S_1 and S_2 indicate the spiral flight movement that is formulated using Eqs. (17) and (18):

$$S_1 = R(i) \times \left(\frac{r_4 \times P(i)}{2\pi}\right) \times cos\big(P(i)\big) \quad (17)$$

$$S_2 = R(i) \times \left(\frac{r_5 \times P(i)}{2\pi}\right) \times sin\big(P(i)\big) \quad (18)$$

Where $R(i)$ is evaluated in Eq. (15) and r_3, r_4, and r_5 represent a random variable with a value ranging from zero to one. Furthermore, $rand_{p_2}$ and $rand_{p_3}$ indicate the random integer in $(0,1)$ to attain the suitable approach in the exploitation stage. Furthermore, other vultures have turned out to be aggressive in the process of foraging If $F < 0.5$ and is assessed by the following expression:

$$P(i+1) = \begin{cases} \frac{A_1 + A_2}{2} & if\ P_3 \geq rand_{p_3} \\ R(i) - |R(i) - P(i)| \times F \times Levy\big(X \times R(i)\big) & if\ P_3 < rand_{p_3} \end{cases} \quad (19)$$

In Eq. (19), A_1 and A_2 denote the vulture movement and are characterized as follows:

$$A_1 = Best\ Vulture_1(i) - \frac{BestVulture_1(i) \times P(i)}{BestVulture_1(i) - P(i)^2} \times F \quad (20)$$

$$A_2 = Best\ Vulture_2(i) - \frac{BestVulture_2(i) \times P(i)}{BestVulture_2(i) - P(i)^2} \times F \quad (21)$$

Moreover, the Levy movement enhances the AVOA approach's effectiveness. The control parameter of AVOA used was fixed as 0.6, 0.4, 0.6, 0.8, and 0.2 for P_1, P_2, P_3, L_1 and L_2, correspondingly. Lastly, the AVOA technique proved to be effective in resolving different kinds of optimization problems. The steps of AVOA are given in the following:

- Define the size of the population and the maximal amount of iterations;
- Calculate the vulture fitness values;
- Choose $R(i)$ by implementing Eq. (15) for each vulture;
- Use Eq. (17) to calculate the location of the optimum vulture;
- Based on the satiation rate of the vulture, Eq. (16) or (19) is used to upgrade the location of the vulture;
- Save the optimum vulture location and then calculate the FF value if the maximum iteration number is not attained.

The AVOA approach derives an FF to accomplish good efficiency of classification. The AVOA approach also determines a positive integer to indicate the greater candidate solution's achievement. At present, the lessening of the classifier error rate can be considered as an FF.

$$fitness(x_i) = ClassifierErrorRate(x_i) = \frac{no.of\ misclassified\ instances}{Total\ no.of\ instances} * 100 \quad (22)$$

4. Performance Validation

In this segment, the investigational validation of the BMFO-DWNMD approach is investigated by employing a benchmark dataset [21]. It involves 10868 samples with nine class labels, namely Ramnit depicting C-1, Lollipop depicting C-2, Kelihos_Ver3 depicting C-3, Vundo depicting C-4, Simda depicting C-5, Tracur depicting C-6, Kelihos_Ver1 depicting C-7, Obfuscator.ACY depicting C-8, and Gatak depicting C-9 correspondingly is depicted below.

Table 1. Dataset details

Labels	No. of Instances
C-1	1541
C-2	2478
C-3	2942
C-4	475
C-5	42
C-6	751
C-7	398
C-8	1228
C-9	1013
Overall Instances	**10868**

The confusion matrices of the BMFO-DWNMD model under malware detection are portrayed in Fig. 3. The outputs portrayed that the BMFO-DWNMD model has recognized all the malware outbreaks competently.

In Table 2 and Fig. 4, the comprehensive malware recognition outputs of the BMFO-DWNMD method are investigated on 80 and 20 percent of TRS/TSS data. The outputs indicated that the BMFO-DWNMD method has been properly classified under both sets. As a sample, with 80% of TRS, the BMFO-DWNMD methodology reaches average $accu_y$ of 97.09%, $sens_y$ of 70.27%, $spec_y$ of 98.25%, F_{score} of 71.67%, MCC of 70.28%, and G_{mean} of 77.67%. Also, with 20% of TSS, the BMFO-DWNMD methodology achieves average $accu_y$ of 97.14%, $sens_y$ of 69.31%, $spec_y$ of 98.27%, F_{score} of 71.18%, MCC of 70.29%, and G_{mean} of 76.91%.

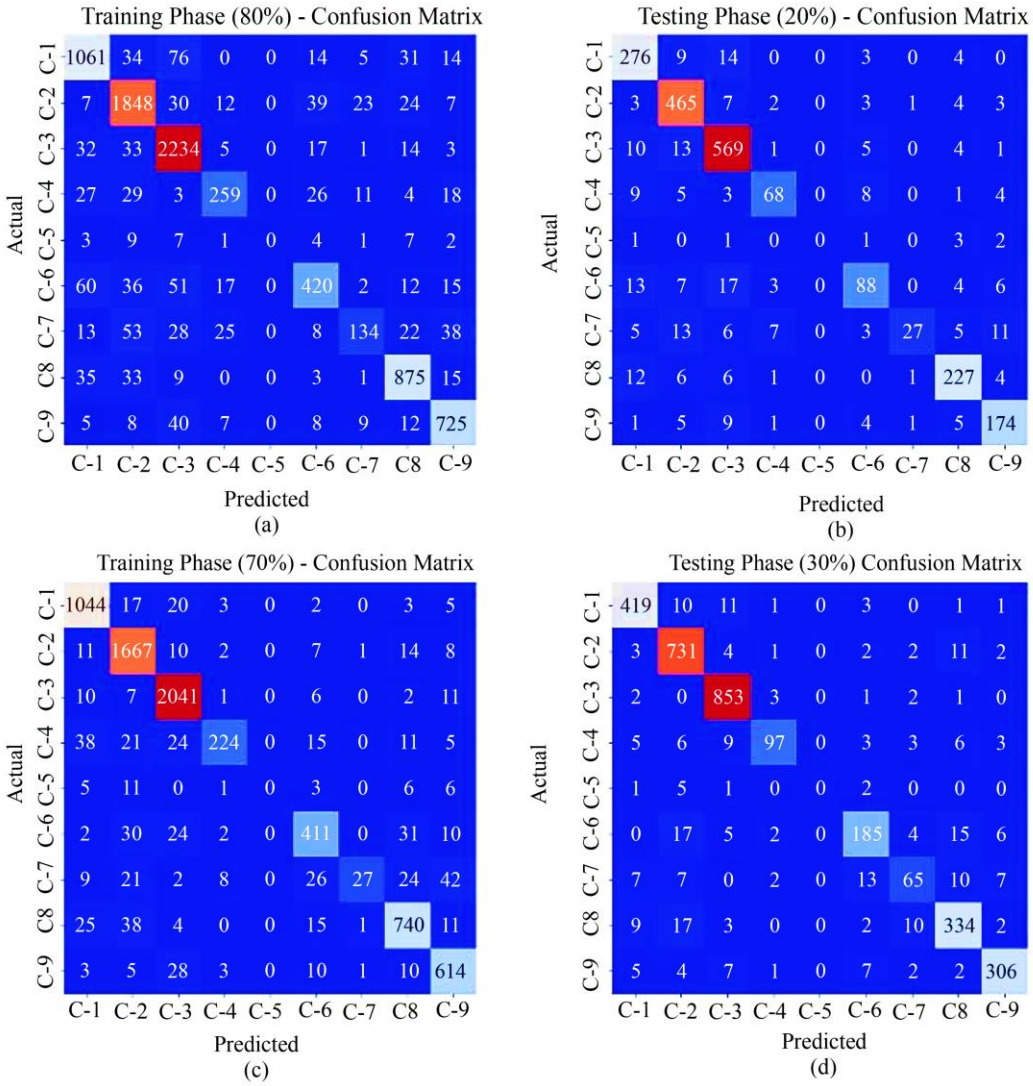

Fig. 3 Confusion matrices of BMFO-DWNMD approach (a-b) and (c-d) 80:20 and 70:30 TRS/TSS

Table 2. Malware identification output of BMFO-DWNMD technique on 80 and 20 percent of TRS/TSS

Labels	Acc_y	$Sens_y$	$Spec_y$	F_{score}	MCC	G_{mean}
Training (80%)						
C-1	95.91	85.91	97.56	85.63	83.25	91.55
C-2	95.66	92.86	96.49	90.74	87.95	94.66
C-3	95.99	95.51	96.16	92.75	90.05	95.84
C-4	97.87	68.70	99.19	73.68	72.79	82.55
C-5	99.61	00.00	100.00	00.00	00.00	00.00
C-6	96.41	68.52	98.53	72.92	71.17	82.16
C-7	97.24	41.74	99.37	52.76	53.44	64.41
C-8	97.45	90.11	98.37	88.74	87.32	94.15
C-9	97.69	89.07	98.58	87.83	86.56	93.70
Average	**97.09**	**70.27**	**98.25**	**71.67**	**70.28**	**77.67**
Testing (20%)						
C-1	96.14	90.20	97.11	86.79	84.62	93.59
C-2	96.32	95.29	96.62	92.08	89.77	95.95
C-3	95.54	94.36	95.99	92.15	89.08	95.17
C-4	97.93	69.39	99.28	75.14	74.34	83.00
C-5	99.63	00.00	100.00	00.00	00.00	00.00
C-6	96.46	63.77	98.67	69.57	68.02	79.32
C-7	97.56	35.06	99.86	50.47	55.33	59.17
C-8	97.24	88.33	98.44	88.33	86.76	93.24
C-9	97.42	87.44	98.43	86.14	84.73	92.77
Average	**97.14**	**69.31**	**98.27**	**71.18**	**70.29**	**76.91**

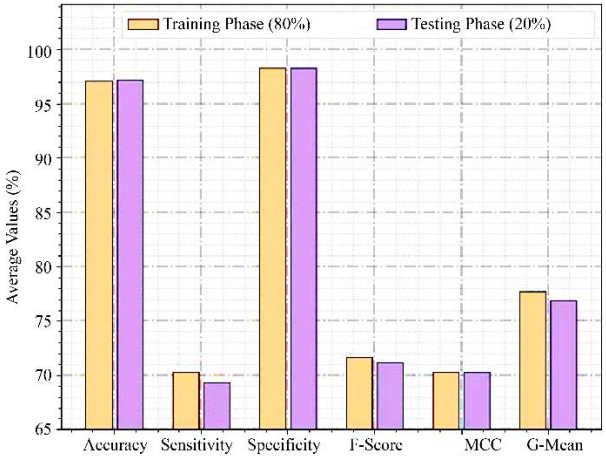

Fig. 4 Malware identification output of BMFO-DWNMD technique on 80 and 20 percent of TRS/TSS

In Table 3 and Fig. 5, the complete malware recognition outputs of the BMFO-DWNMD approach are inspected on 70 and 30 percent of TRS/TSS data. The outputs specified that the BMFO-DWNMD system has accurately classified under both sets. For example, with 70% of TRS, the BMFO-DWNMD system reaches average $accu_y$ of 97.92%, $sens_y$ of 74.05%, $spec_y$ of 98.77%, F_{score} of 75.67%, MCC of 74.78%, and G_{mean} of 80.18%. Furthermore, with 30% of TSS, the BMFO-DWNMD system achieves average $accu_y$ of 98.15%, $sens_y$ of 75.66%, $spec_y$ of 98.90%, F_{score} of 77.11%, MCC of 76.18%, and G_{mean} of 81.23%.

Table 3. Malware identification output of BMFO-DWNMD technique on 70 and 30 percent of TRS/TSS

Labels	Acc_y	$Sens_y$	$Spec_y$	F_{score}	MCC	G_{mean}
Training (70%)						
C-1	97.98	95.34	98.42	93.13	91.98	96.87
C-2	97.31	96.81	97.45	94.21	92.51	97.13
C-3	98.01	98.12	97.97	96.43	95.09	98.05
C-4	98.17	65.31	99.72	76.32	76.58	80.70
C-5	99.57	00.00	100.00	00.00	00.00	00.00
C-6	97.50	79.50	98.82	81.23	79.91	88.63
C-7	97.70	54.01	99.41	63.92	63.93	73.27
C-8	97.21	86.96	98.51	87.47	85.90	92.55
C-9	97.86	90.43	98.59	88.28	87.13	94.42
Average	**97.92**	**74.05**	**98.77**	**75.67**	**74.78**	**80.18**
Testing (30%)						
C-1	98.19	93.95	98.86	93.42	92.38	96.37
C-2	97.21	96.69	97.37	94.14	92.37	97.03
C-3	98.50	98.96	98.33	97.21	96.21	98.64
C-4	98.62	73.48	99.68	81.17	80.94	85.59
C-5	99.72	00.00	100.00	00.00	00.00	00.00
C-6	97.49	79.06	98.91	81.86	80.57	88.43
C-7	97.88	58.56	99.27	65.33	64.71	76.24
C-8	97.27	88.59	98.40	88.24	86.70	93.37
C-9	98.50	91.62	99.28	92.59	91.76	95.37
Average	**98.15**	**75.66**	**98.90**	**77.11**	**76.18**	**81.23**

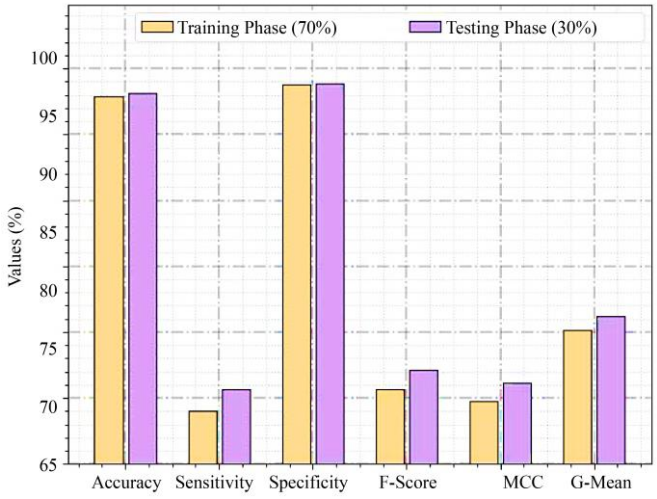

Fig. 5 Malware identification output of BMFO-DWNMD technique on 70 and 30 percent of TRS/TSS

The TACC value and VACC value of the BMFO-DWNMD methodology are tested on cyberattack achievement in Fig. 6. The figure is implicit that the BMFO-DWNMD methodology has depicted an improved achievement with greater TACC and VACC values. It is evident that the BMFO-DWNMD approach has attained greater TACC outputs.

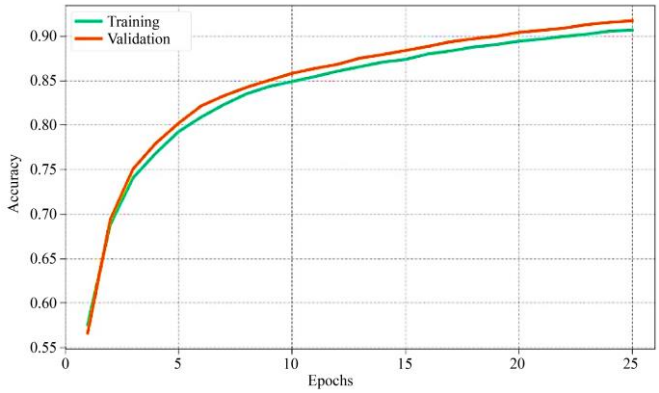

Fig. 6 TACC and VACC output of BMFO-DWNMD technique

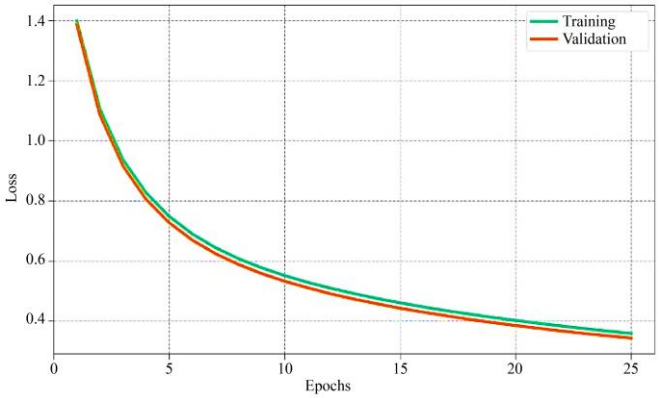

Fig. 7 TLS and VLS output of BMFO-DWNMD approach

The TLS value and VLS value of the BMFO-DWNMD technique have experimented on cyberattack achievement in Fig. 7. The figure indicated that the BMFO-DWNMD approach had depicted augmented achievement with lesser TLS and VLS values. It is evident that the BMFO-DWNMD approach has reached a minimum VLS output.

A precise precision-recall inspection of the BMFO-DWNMD system under the testing data is depicted in Fig. 8. The outputs stated that the system has given an outcome in greater values under total classes.

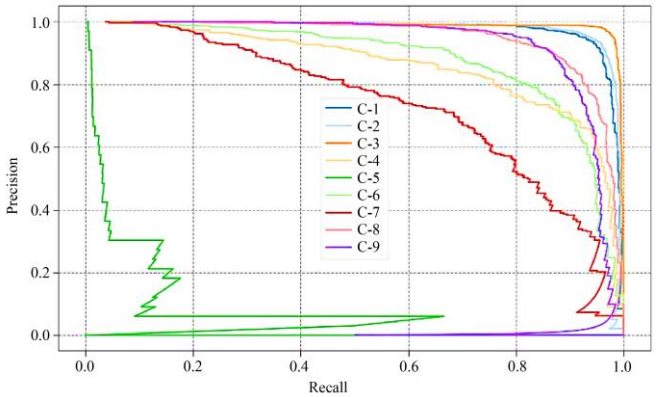

Fig. 8 Precision-recall output of BMFO-DWNMD approach

A short ROC evaluation of the BMFO-DWNMD technique under the testing data is described in Fig. 9. The outputs specified that the BMFO-DWNMD technique has shown its capacity to classify distinct classes.

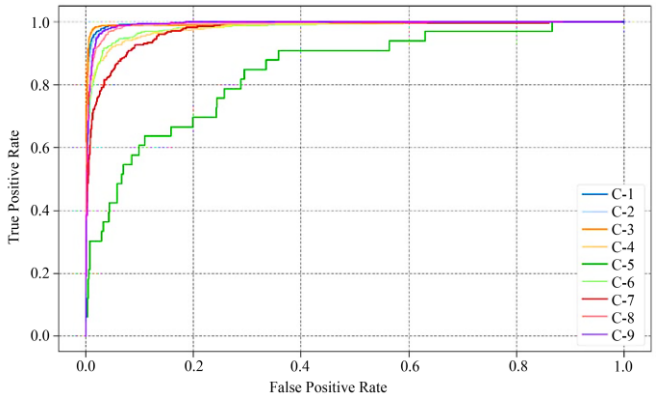

Fig. 9 ROC output of BMFO-DWNMD approach

To ensure an enhanced achievement of the BMFO-DWNMD model, far-reaching relative research is accomplished in Table 4 [22]. Fig. 10 exhibits a comparison research of the BMFO-DWNMD model over other techniques by means of $accu_y$. The investigational outputs notified that the BMFO-DWNMD approach had better performance. It is noticed that the BMFO-DWNMD approach attains a higher $accu_y$ of 98.15%, whereas the linear-SVM, RBF-SVM, MLP,

CNN, and DLMD techniques obtain lower $accu_y$ of 96.82%, 93.82%, 97.18%, 95.62%, and 95.21% subsequently.

Table 4. Relative evaluation of the BMFO-DWNMD method with other approaches

Methods	Accuracy (%)	Computational Time (sec)
BMFO-DWNMD	98.15	6.46
Linear-SVM	96.82	14.23
RBF-SVM	93.82	12.69
MLP	97.18	18.2
CNN	95.62	9.62
DLMD	95.21	18.42

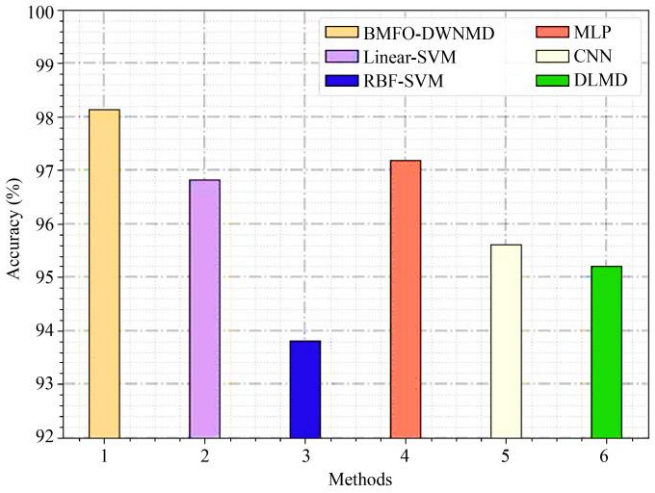

Fig. 10 $Accu_y$ evaluation of the BMFO-DWNMD method with other approaches

Fig. 11 shows a comparative investigation of the BMFO-DWNMD method with other methods in terms of CT. The simulation outcomes illustrated that the BMFO-DWNMD method attains effective results with a minimal CT of 6.46s. On the other hand, the linear-SVM, RBF-SVM, MLP, CNN, and DLMD technique accomplishes lower CT of $14.23s, 12.69s, 18.2s, 9.62s$,and $18.42s$ subsequently.These outputs show the enhanced achievement of the proposed technique for malware detection.

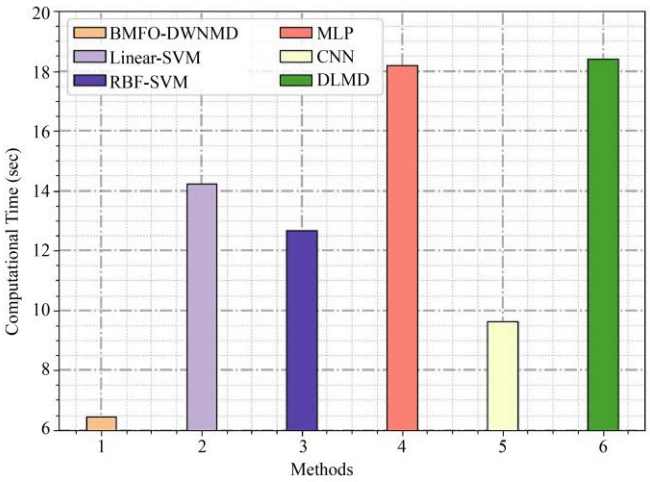

Fig. 11 CT analysis of BMFO-DWNMD approach with other approaches

5. Conclusion

This article uses a novel BMFO-DWNMD approach for automatic malware detection to accomplish cybersecurity. The presented BMFO-DWNMD technique focuses on the recognition and classification of malware using classification and FS process. In the proposed BMFO-DWNMD approach, the BMFO approach is exploited for the optimum FSB selection. Next, the BMFO-DWNMD model uses a DWN classifier to recognize malware attacks. Lastly, the AVOA is exploited for the process of hyperparameter tuning. A comprehensive set of simulations has been performed to depict the investigational validation of the BMFO-DWNMD model. The experimental outcomes illustrate an enhanced achievement of the BMFO-DWNMD model over other models. In the future, the data clustering process can be exploited to reduce the computation complexity of the BMFO-DWNMD technique.

References

[1] Suryakant Badde et al., "Cyber Attack Detection Framework for Cloud Computing," *Intelligent Data Engineering and Analytics*, vol. 1177, pp. 243-254, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[2] Tran Viet Khoa et al., "Collaborative Learning Model for Cyberattack Detection Systems in IoT Industry 4.0," *2020 IEEE Wireless Communications and Networking Conference*, pp. 1-6, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[3] Truong Thu Huong et al., "Lockedge: Low-Complexity Cyberattack Detection in IoT Edge Computing," *IEEE Access*, vol. 9, pp. 29696-29710, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[4] Fargana J. Abdullayeva, "Detection of Cyberattacks in Cloud Computing Service Delivery Models using Correlation Based Feature Selection," *2021 IEEE 15th International Conference on Application of Information and Communication Technologies*, pp. 1-4, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Rahul Chourasiya, Vaibhav Patel, and Anurag Shrivastava, "Classification of Cyber Attack using Machine Learning Technique at Microsoft Azure Cloud," *International Research of Engineering and Applied Sciences*, vol. 6, no. 1, pp. 4-8, 2018. [Google Scholar] [Publisher Link]

[6] Prabhat Kumar, Govind P. Gupta, and Rakesh Tripathi, "An Ensemble Learning and Fog-Cloud Architecture-Driven Cyber-Attack Detection Framework for IoMT Networks," *Computer Communications*, vol. 166, pp. 110-124, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[7] Carmelo Ardito et al., "An Artificial Intelligence Cyberattack Detection System to Improve Threat Reaction in e-Health," *ITASEC*, vol. 2940, pp. 270-283, 2021. [Google Scholar] [Publisher Link]

[8] MD. Mamunur Rashid et al., "Cyberattacks Detection in IOT-Based Smart City Applications using Machine Learning Techniques," *International Journal of Environmental Research and Public Health*, vol. 17, no. 24, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[9] Khoi Khac Nguyen et al., "Cyberattack Detection in Mobile Cloud Computing: A Deep Learning Approach," *2018 IEEE Wireless Communications and Networking Conference*, pp. 1-6, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[10] Xiayang Chen et al., "Ensemble Learning Methods for Power System Cyber-Attack Detection," *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis*, pp. 613-616, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[11] Abdulrahman Al-Abassi et al., "An Ensemble Deep Learning-Based Cyber-Attack Detection in Industrial Control System," *IEEE Access*, vol. 8, pp. 83965-83973, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[12] Iqbal H. Sarker et al., "IntruDTree: A Machine Learning Based Cyber Security Intrusion Detection Model," *Symmetry*, vol. 12, no. 5, pp. 1-15, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] Yihao Wan, and Tomislav Dragičević, "Data-Driven Cyber-Attack Detection of Intelligent Attacks in Islanded DC Microgrids," *IEEE Transactions on Industrial Electronics*, vol. 70, no. 4, pp. 4293-4299, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[14] Qasem Abu Al-Haija, and Saleh Zein-Sab, "An Efficient Deep-Learning-Based Detection and Classification System for Cyber-Attacks in IoT Communication Networks," *Electronics*, vol. 9, no. 12, pp. 1-26, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[15] Mahmoud Elsisi et al., "Towards Secured Online Monitoring for Digitalized GIS Against Cyber-Attacks Based on IoT and Machine Learning," *IEEE Access*, vol. 9, pp. 78415-78427, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[16] Ahmad Ali AlZubi, Mohammed Al-Maitah, and Abdulaziz Alarifi, "Cyber-Attack Detection in Healthcare using Cyber-Physical System and Machine Learning Techniques," *Soft Computing*, vol. 25, no. 18, pp. 12319-12332, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[17] S. Kannan, and T. Pushparaj, "Creation of Testbed Security using Cyber-Attacks," *SSRG International Journal of Computer Science and Engineering*, vol. 4, no. 11, pp. 4-14, 2017. [CrossRef] [Publisher Link]

[18] Ruba Abu Khurma, Ibrahim Aljarah, and Ahmad Sharieh, "A Simultaneous Moth Flame Optimizer Feature Selection Approach Based on Levy Flight and Selection Operators for Medical Diagnosis," *Arabian Journal for Science and Engineering*, vol. 46, no. 9, pp. 8415-8440, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[19] Salwa Said et al., "Deep Wavelet Network for Image Classification," *2016 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 922-927, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[20] Ahmed Hossam-Eldin et al., "Improving the Frequency Response of Hybrid Microgrid under Renewable Sources' Uncertainties using a Robust LFC-Based African Vulture Optimization Algorithm," *Processes*, vol. 10, no. 11, pp. 1-19, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[21] Mansour Ahmadi et al., "Novel Feature Extraction, Selection and Fusion for Effective Malware Family Classification," *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*, pp. 183-194, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[22] Muhammad Furqan Rafique et al., "Malware Classification using Deep Learning Based Feature Extraction and Wrapper Based Feature Selection Technique," *arXiv*, pp. 1-21, 2019. [CrossRef] [Google Scholar] [Publisher Link]