IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 June 2024, accepted 3 July 2024, date of publication 9 July 2024, date of current version 18 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3425593

RESEARCH ARTICLE

Enhanced Image-Based Malware Classification Using Snake Optimization Algorithm With Deep Convolutional Neural Network

SALAHALDEEN DURAIBI
College of Engineering and Computer Science, Jazan University, Jazan 45142, Saudi Arabia
e-mail: sduraibi@jazanu.edu.sa

The author gratefully acknowledge the funding of the Deanship of Graduate Studies and Scientific Research, Jazan University, Saudi Arabia, through Project Number: GSSRD-24.

ABSTRACT Malware is a malicious software intended to cause damage to computer systems. In recent times, significant proliferation of malware utilized for illegal and malicious goals has been recorded. Several machine and deep learning methods are widely used for the detection and classification of malwares. Image-based malware detection includes the usage of machine learning and computer vision models for analyzing the visual representation of malware, including binary images or screenshots, for the purpose of detecting malicious behaviors. This techniques provides the potential to identify previously hidden or polymorphic malware variants based on the visual features, which provide a further layer of defense against emerging cyber-attacks. This study introduces a new Snake Optimization Algorithm with Deep Convolutional Neural Network for Image-Based Malware Classification technique. The primary intention of the proposed technique is to apply a hyperparameter-tuned deep learning method for identifying and classifying malware images. Primarily, the ShuffleNet method is mainly used to derivate the feature vectors. Besides, the snake optimization algorithm can be deployed to boost the choice of hyperparameters for the ShuffleNet algorithm. For the recognition and classification of malware images, attention-based bi-directional long short-term memory model. The simulation evaluation of the proposed algorithm has been examined using the Malimg malware dataset. The experimental values inferred that the proposed methodology achieves promising performance with a maximum accuracy of 98.42% compared to existing models.

INDEX TERMS Malware detection, Snake Optimization Algorithm, deep learning, ShuffleNet, convolutional neural network.

I. INTRODUCTION

Malware is malicious software that evolved purposefully and affects computer systems [1]. It is utilized to infiltrate, attack, or acquire accessibility to any digital resources that can be highly complex and cause loss or undesirable outcomes in a system [2]. The main aim is to cause damage and attack resources whose accessibility is not open. Approximately 360,000 novel malware files have been identified regularly, and the quantity of files created every day must be raised by 5.2% [3]. The rapid evolution and widespread dissemination of malware are made possible by using automated and sophisticated malware creation tools [4]. Anti-virus software depends upon two frequently employed methods, behaviour- and signature-based detection, to identify and classify malware [5]. The signatures of the malware have been generally gained from known malware by executing static analysis (no implementation of the malware) [6]. A database made of signatures gathered from different malicious objects that could be employed for malware identification and classification [7]. Although the signature-based detection technique is extremely fast and accurate, it can be simply avoided by using obfuscation algorithms (for example, metamorphism, packing, encryption, and polymorphism) to produce a new variant [8].

Furthermore, the signature database, typically reliant on static analysis, has traditionally been manually updated and curated, resulting in labour-intensive and time-consuming

The associate editor coordinating the review of this manuscript and approving it for publication was Yassine Maleh.

processes [9]. Unlike signature-based detection, in behaviour-based detection, behaviours of the malware can be obtained and recorded in the execution of specified malicious codes on dynamic analysis [10]. Therefore, this method will recognize metamorphic and polymorphic viruses dependent upon their behaviours. However, storing run-time behavioural patterns is deliberated by extensive resources [11]. Additionally, the behaviour dataset must be upgraded once a new malware type can be determined. An innovative technique is of great importance for malware classification, which depends on image processing [12]. According to the texture of the malware image, this algorithm permits a classifier to identify and categorize the present malware samples that could be transformed from the collected malware binary.

Various researchers have developed numerous innovative methods, a few of which are accomplished by employing the previous techniques for classification [13]. Primarily, shallow machine learning (ML) methods are implemented for the classification of malware instances. Convolution neural networks (CNNs) entered into presence in the '90s [14]. They became extensively utilized as a region of interest (RoI) for computer vision (CV) relevant tasks, particularly images, speech, and time series [15]. Similarly, it can be highly efficacious in object detection for identification, classification difficulties, and so on. Stimulated by the achievement of CNN in classification difficulties, several studies have been implemented to understand the usage of CNN in malware classification [16]. CNN is the deep neural network (DNN) that exceedingly depends on convolutional methods and comprises layers such as a fully connected (FC) layer, pooling layer, and convolution layer, which are developed for automatically and flexibly learning the spatial hierarchies of features by employing a backpropagation (BP) method [17]. The CNN model automated the feature extraction and entwined learned features with input data, creating a robust tool for classifying information with vast amounts of features or images [18].

This study introduces a new Snake Optimization Algorithm with Deep Convolutional Neural Network for Image-Based Malware Classification (SODCNN-IMC) technique. In the SODCNN-IMC model, the ShuffleNet technique is applied for effectual derivation of the feature vectors. Besides, the SO algorithm could be exploited to boost the choice of hyperparameters in the ShuffleNet architecture. For the detection and classification of malware images, attention-based bi-directional long short-term memory (ABiLSTM) approach. The performance evaluation of the SODCNN-IMC technique is validated using the Malimg malware dataset. The experimental values inferred that the SODCNN-IMC methodology achieves excellent performance over other methods in terms of diverse evaluation measures. In short, the key contributions of the study is given as follows:

- Design a strong malware detection using SODCNN-IMC method through efficiently leveraging the feature extracted by the ShuffleNet, finetuned the hyperparameter through the Snake Optimization, and the attention module of ABiLSTM.
- Utilizes ShuffleNet for effective feature extraction from images that considerably decreases the computation difficulty while maintaining superior performance, making it fit for resource-constraint environment.
- Develops SO technique to automatically finetune the hyperparameter of the ShuffleNet model that effectively explore the hyperparameter space, which enables the model to adapt and enhance its performance for various datasets.
- Applies BiLSTM layers for capturing long-range dependencies and focus on significant parts within the input images. The attention module improves the model's capability to distinguish between benign and malicious features, enhancing the overall accuracy of detection.
- Hyperparameter optimization of the ABiLSTM algorithm using SO technique based cross-validation assist in boosting the prediction outcomes of the presented technique for hidden data.

The rest of the paper is organized as follows. Section II provides the related works and section III offers the proposed model. Then, section IV gives the result analysis and section V concludes the paper.

II. RELATED WORKS

Zhao et al. [19] presented a static malware identification method dependent upon the AlexNet CNN model. Instead of present solutions, the developed method converted each malicious byte into colour images. It also provides an enhanced AlexNet framework and resolves the unbalanced databases with the data improvement algorithm. In [20], a new visual malware recognition architecture dependent upon DNNs was developed. Initially, execution file instances were composed and transformed into bytes and asm files using disassembled technology. Next, visualization technology integrated with data augmentation was utilized for additional sample conversion into 3-channel RGB images. Lastly, a DNN model was introduced, i.e. SEResNet50 + Bi-LSTM + Attention (SERLA). Chaganti, Ravi and Pham [21] projected an effective NN architecture, EfficientNetB1, employing the level of malware byte image representation method. To mitigate the computational resources and consumption caused by DL methods testing and training the different CNN-based techniques, the technique executed the task and computational efficacy assessment of the diverse CNN pre-trained architecture for choosing the preeminent CNN model for classifying malware.

In [22], a new DeepGray technique was presented to classify multi-class malware through grayscale images and the supremacy of DL. This method includes converting execution files into a format appropriate for DL. In the data preprocessing stage, Principal Component Analysis (PCA) was implemented. ViT, VGG16, EfficientNet, and modified CNN frameworks could be employed for classification. Bakour and

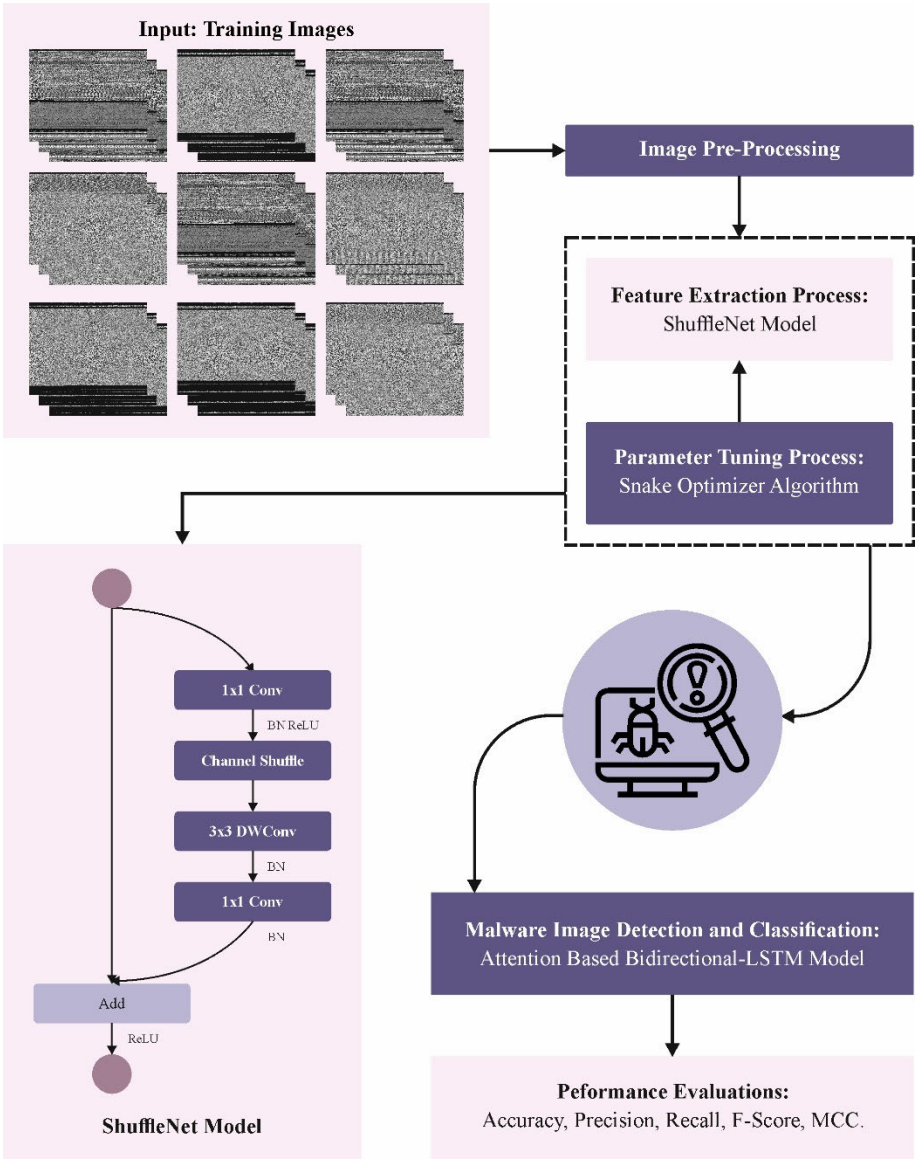

FIGURE 1. Overall procedure of the SODCNN-IMC algorithm.

Ünver [23] implemented an innovative hybrid DL algorithm named DeepVisDroid. There are two kinds of image-based features, such as global and local features, that must be removed. Subsequently, 1D convolutional layers-based NN architecture was designed and trained. Additionally, two standard 2D convolutional layers-based NN frameworks could be developed, and two well-known DL methods were examined.

In [24], a lightweight malware classification method named IMCLNet was introduced, which must be determined by malware images and required domain knowledge and feature engineering. While developing the architecture, the method widely weighed accuracy, multiple parameters, and computational rate and incorporated Coordinate Attention, Global Context Embedding, and Depth-wise Separable Convolution.

Copiaco et al. [25] designed a new multi-functional malware recognition architecture. This method discovers numerous pre-trained networks comprising traditional and compact networks, series, and fixed acyclic graph structures for classifying malware. The algorithm exploits grayscale transform-based features as consistent features, converting malware classification through different file categories. The technique incorporates several databases into the training model. In [26], an innovative method employing image-based DL classification space. Especially, Jadeite removes the Inter-procedural Control Flow Graph (ICFG) from a specific Java bytecode file followed by clipping the ICFG and transforming it into an adjacent matrix. The architecture leverages an object identification technique in a DCNN model to detect maliciousness.

In [27], a multi-headed attention based technique is combined to a CNN to find and classify the tiny diseased areas in the complete image. The performance of the planned multi-headed attention-based CNN technique was equated with numerous non-attention-CNN-based techniques on numerous data splits of testing and training malware image benchmark dataset. Chaganti et al. [28] proposed a DL-based CNN method in order to achieve the malware identification on Portable Executable (PE) dual files utilizing the fusion feature set model. We presented a wide performance assessment of numerous DL method structure and ML classifier i.e. Support Vector Machine (SVM), on multi-aspect feature sets covering the dynamic, static, and image features to pick the developed CNN method. Reilly et al. [4] explores the efficiency of training DL methods with Generative Adversarial Network-generated data to recover their sturdiness beside such assaults. Ben Abdel Ouahab et al. [29] developed and test a malware classifier capable to affect every inputted malware into its equivalent family. To do so, we utilize the multi-layer perceptron technique with malware visualization model.

III. THE PROPOSED METHOD

In this research, we have developed an innovative SODCNN-IMC system. The foremost goal of the SODCNN-IMC model is to apply a hyperparameter-tuned DL method for recognizing and categorizing malware images. The SODCNN-IMC technique contains three major processes, namely ShuffleNet-based feature extraction, SO-based parameter tuning, and ABiLSTM-based classification process. Fig. 1 demonstrates the complete procedure of the SODCNN-IMC system.

A. FEATURE EXTRACTION

ShuffleNet architecture can be applied to the effectual derivation of feature vectors for feature extraction. We implement the ShuffleNet, an extremely effective DL model produced with mobile devices [30]. According to the computation resource (hardware), we applied the shufflenetV1 version of the pre-trained ShuffleNet method to obtain the best outcome at a lower computation cost. This introduced technique can be higher than typical CNN with 50 learnable layers through the FC layer, one convolutional (Conv) layer, and 48 group Conv layers. The CNN model has a total of 172 layers, involving 49 BN layers, a classification layer, one max pooling layer, four average pooling layers, a softmax layer, and 33 ReLu layers.

The primary layer is the input layer and accepts an image size of 224×224 (CT scan, ECG trace image, or chest radiograph) [31]. The first Conv layer was utilized for removing features in the input image of 224×224 at 24 filters (kernels) of 3×3 size with a stride of 2×2 to generate the feature map. The outcome of the Conv layer is evaluated by:

$$s(i,j) = (I \times K)(i,j) = \sum_{n} \sum_{m} I(m,n) K(i-m, j-n) \quad (1)$$

In Eq. (1), s denotes the outcome mapping feature, i indicates the input image, *and* K shows the filter of the Conv layer. The output of size $o = ((i-k)+2p)/(s+1)$ will be produced later employing the Conv process with the input image, whereas i denotes input, p implies padding, s signifies steps, and k means the size of kernels.

The ShuffleNet architecture with stride (shift) of 2×2 receives the resultant mapping features of the 1st Conv layer. Its component contains three Conv procedures, viz., 3×3 depthwise Convs and dual 1×1 pointwise group Conv. A primary point-wise group Conv has been proceeded by *BN*, channel shuffle process, and ReLU activation function. ReLU activation is used since it is straightforward and efficient.

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \quad (2)$$

ReLU stimulates neurons with positive value and disables neurons with negative value of 0. The architecture has a 3-by-3 average pooling under the shorter path. It has 16 consecutive ShuffleNet elements. Also, it has 50 layers, all are contained the trained feature map [32]. Also, the layer performs feature extraction. Softmax activation could be employed to resolve the classification possibility employed by the ultimate classification layer.

$$a_i = \sum_{j=0}^{m \times n - 1} w_{ij} \times x + b_i \quad (3)$$

In Eq. (3), b denotes the bias; w represents the weights. i refers to the index output of the FC layer; i, m, d, and n are the index, width, depth, and height of the FC layer outcome. The classification probability of the softmax layer could be generated up to 1000 distinct classes.

B. HYPERPARAMETER TUNING USING SO MODEL

Besides, the SO algorithm can be exploited to select the optimum hyperparameter for the ShuffleNet architecture. SO has been developed to handle a diverse set of optimization functions that reproduce the superior mating behavior of snakes [33]. All snakes (male/female) compete to have the most significant partner once the present quantity of food is sufficient and the temperature can be lower. The SO architecture depends on two main phases similar to the alternative swarm-based methods: local search (exploitation) and global search (exploration). While food is rare and the conditions are formal, snakes in the exploration stage are distributed through the hunting region. Alternatively, the exploitation phase was broken down into numerous smaller phases. This part clarifies the arithmetical method of the SO algorithm. Thus, the below-mentioned steps demonstrate the SO method:

Initialize the solutions: SO begins a range of random solutions by employing Eq. (4) in the search space. These outcomes form the snake population of SO will be enhanced

in the following steps:

$$S_i = S_{\min} + R \times (S_{\max} - S_{\min}) \tag{4}$$

where S_i means the place of the *ith* snake. An arbitrary number in [0 *and* 1] is named R. The maximum and minimum potential values are $S_{\max}$ and $S_{\min}$, correspondingly

Snake population separation: Utilizing Eqs. (5) and (6), the population is divided equally into dual parts like 50 per cent female and 50 per cent male:

$$N_m = \frac{N}{2} \tag{5}$$

$$N_f \approx N - N_m \tag{6}$$

whereas N denotes the total population size. N_f and N_m represent the total amount of female and male snakes.

Estimation of snakes: Acquire the ideal snake from the female and male groups (S_{bestm} and S_{bestf}), and discover the food (L_{food}). The descriptions of dual extra terms describe food quantity (Q) and temperature (T), which are stated separately as Eqs. (7) and (8):

$$T = e^{\frac{-c}{C}} \tag{7}$$

where c denotes the present iteration, and C represents the maximal amount of iterations.

$$Q = k_1 x \ominus \left(\frac{c - C}{c}\right) \tag{8}$$

where K_1 represents the constant value equivalent to 0.5.

Exploring the searching region (food is not determined): This mainly depends upon the usage of a selected value of the threshold. When Q is 0.25, the snakes upgrade their locations relative to an assumed arbitrary place to hunt globally. It is defined in Eqs. (9) to (12).

$$S_{mi}(c+1) = S_{mR}(c) \pm K_2 \times AB_m \times ((S_{\max} - S_{\min}) \times R + S_{\min}) \tag{9}$$

where R denotes the random amount in [0 *and* 1], S_{mi} and S_{mR} refer to the place of i^{th} and arbitrary male snake. The ability of the male snake to detect food can be signified by AB_m and exposed utilizing Eq. (7):

$$AB_m = e^{(-F_{mR})/(F_{mi})} \tag{10}$$

whereas K_2 denotes the constant equivalent to 0.05, F_{mR} refers to the fitness of the S_{mR} snake, and F_{mi} signifies the fitness of the i^{th} snake from the male group.

$$S_{fi}(c+1) = S_{fR}(c) \pm K_2 \times AB_f \times ((S_{\max} - S_{\min}) \times R + S_{\min}) \tag{11}$$

where S_{fR} denotes the random female snake place, R indicates the random number at 0 and 1, S_{fi} stands for the female snake place with i^{th}, and AB_f is the capability of the female snake to discover foodstuff.

$$AB_f = e^{(-F_{fR})/(F_{fi})} \tag{12}$$

where F_{fR} signifies the fitness of the female snake group S_{fR}, F_{fi} represents the fitness of the snake within the *ith* snake, and K_2 is a constant equivalent to 0.05.

Exploiting the searching region (food is found): The temperature should be verified when the quantity of food is more significant than a pre-determined threshold $Q > 0.25$. The results can be moved to the food where $T > 0.6$ (hot).

$$S_{(i,j)}(c+1) = L_{food} \pm K_3 \times T \times R \times \left(L_{food} - S_{(i,j)}(c)\right) \tag{13}$$

While $S_{(i,j)}$ epitomizes a snake's location, L_{food} signifies the finest snake, and K3 denotes a constant value equivalent to 2 [34].

The snake can be in the mating or fighting mode if $T > 0.6$ (cold),

$$S_{mi}(c+1) = S_{mi}(c) \pm K_3 \times FA_m \times R \times \left(S_{fbest} - S_{mi}(c)\right) \tag{14}$$

where S_{mi} is the *ith* male location, S_{fbest} denotes the location of the finest snake within the female group, and FA_m signifies the fighting capability of the male snake.

$$S_{fi}(c+1) = S_{fi}(c) \pm K_3 \times FA_f \times R \times \left(S_{mbest} - S_{fi}(c)\right) \tag{15}$$

where S_{fi} implies the i^{th} female place, S_{mbest} denotes the top snake position within the male group, and FA_f represents the fighting capability of the female snake. FA_m and FA_f are originated as mentioned formula:

$$FA_m = e^{(-F_{fbest})/(F_i)} \tag{16}$$

$$FA_f = e^{(-F_{mbest})/(F_i)} \tag{17}$$

where F_{mbest} indicates the best male snake fitness, F_i represents the i^{th} snake fitness, and F_{fbest} designates the best female snake fitness.

$$S_{mi}(c+1) = S_{mi}(c) \pm K_3 \times MA_m \times R \times \left(Q \times S_{fi}(c) - S_{mi}(c)\right) \tag{18}$$

$$S_{fi}(c+1) = S_{fi}(c) \pm K_3 \times MA_f \times R \times \left(Q \times S_{mi}(c) - S_{fi}(c)\right) \tag{19}$$

where S_{fi} and S_{mi} denote the female and male positions of the *ith* snake, correspondingly, and MA_m and MA_f define the abilities of males and females for mating, separately and resultant as below:

$$MA_m = e^{(-F_{fi})/(F_{mi})} \tag{20}$$

$$MA_f = e^{(-F_{mi})/(F_{fi})} \tag{21}$$

If an egg hatches, pick the worst female and male snakes and exchange them.

$$S_{mworst} = S_{\min} + R \times (S_{\max} - S_{\min}) \tag{22}$$

$$S_{fworst} = S_{\min} + R \times (S_{\max} - S_{\min}) \tag{23}$$

S_{fworst} and S_{mworst} denote the worst snakes from the female and male groups, respectively [35]. The *pm* is a diversity

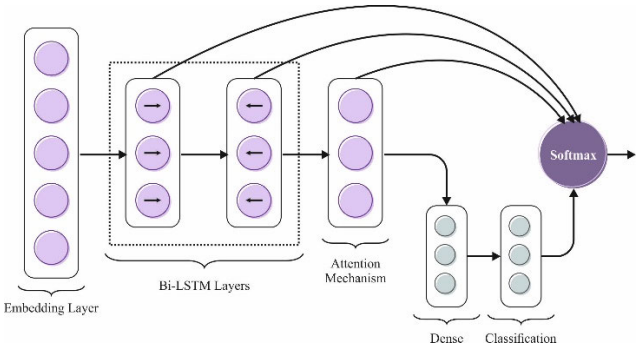

FIGURE 2. Architecture of ABiLSTM model.

factor operator that offers the choice of increasing or reducing snake locations. It is employed to differentiate the snake locations from the search space in any way.

The SO method technique derives a fitness function to attain improved classification performance. It determines a positive integer to represent the better performance of the candidate solutions. In this study, the minimization of the classification error rate is considered as the fitness function as specified in Eq. (24).

$$fitness\,(x_i) = ClassifierErrorRate\,(x_i) = \frac{No.of\ misclassified\ instances}{Total\ no.of\ instances} * 100 \quad (24)$$

C. ABiLSTM-BASED CLASSIFICATION

Eventually, the SODCNN-IMC approach can be utilized in the ABiLSTM model. Like LSTM and Bi-LSTM, a deep neural network (DNN) is introduced for extracting temporal features between network packets since network traffic is an incessant flow of sequence data in packet bytes [36]. LSTM is a variant of RNN used to address explosion and vanishing gradient problems. When equated to RNN, LSTM networks have been significantly more effective in detecting long-term dependency within sequence data, making them the best alternative to extract temporal features from traffic network data. The architecture of LSTM has forget, input, and output gates. Fig. 2 signifies the framework of the ABiLSTM model.

A forget gate is used to control how much data is to be retained or discarded. The data from the prior HLs and the existing input are passed over the sigmoid function that produces values between 0 and 1. When the value is closer to 1, then it is highly possible to be remembered; when the value is closer to zero, then it is highly possible to be forgotten. Eq. (25) evaluates the forget vector, where the parameters W_f and b_f are the forget gates, x_T denotes the input vector at the t step, and the HL vector at step $t-1$ is h_{t-1}.

$$f_t = \sigma\left(W_f.[h_{t-1}, x_t] + b_f\right) \quad (25)$$

The input gate acts as a value evaluator for information included in extended memory as original data. It handles how much x_t input dataset is added to C_t. Also, it defines what information to be forgotten or retained for the cell state [37]. When the value is closer to zero, then more information is discarded; when the input vector value is closer to one, then additional information is retained in long-range memory. W_i and b_i are the input gate parameters.

$$i_t = \sigma\,(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (26)$$

The forgot gate f_t is multiplied by C_{t-1} (the prior cell layer vector); later, the input gate vector is multiplied utilizing $\tilde{C}_t$ point-wise multiplication, $\tilde{C}_t$ indicates the data enclosed from the HL vector.

$$\tilde{C}_t = tanh\,(W_c.[h_{t-1}, x_t] + b_c) \quad (27)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (28)$$

The output gate in LSTM defines what long-term memory must be transmitted to the outcome. W_o and b_o are the output gate parameters.

$$o_t = \sigma\,(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (29)$$

$$h_t = o_t * tanh\,(C_t) \quad (30)$$

BiLSTM is a variant of the sequence processing model composed of 2 LSTMs, namely the forward and reverse direction [38]. The LSTM network is used to address the long-term dependency problem. Due to its architecture, LSTM can memorize data over time and learn long-term information. Compared to LSTM, this study exploits the BiLSTM model due to its high prediction accuracy. The attention module focuses on the data generated by the HL of BiLSTM.

TABLE 1. Details of the dataset.

Classes	No. of Instances
Adialer. C	122
Agent.FYI	116
Autorun. K	106
C2LOP.P	146
Dialplatform. B	177
Dontovo. A	162
Fakerean	381
Lolyda.AA1	213
Rbot!gen	158
Swizzor.gen!E	128
Total Instances	**1709**

IV. PERFORMANCE VALIDATION

The experimental evaluation of the SODCNN-IMC system is validated by employing the Malimg malware database [39]. The database encompasses 1709 samples under ten classes, as described in Table 1. The Malimg dataset is a commonly employed benchmark database in the area of computer security and malware analysis. It includes a collection of grayscale imageries demonstrating dissimilar kinds of malware samples

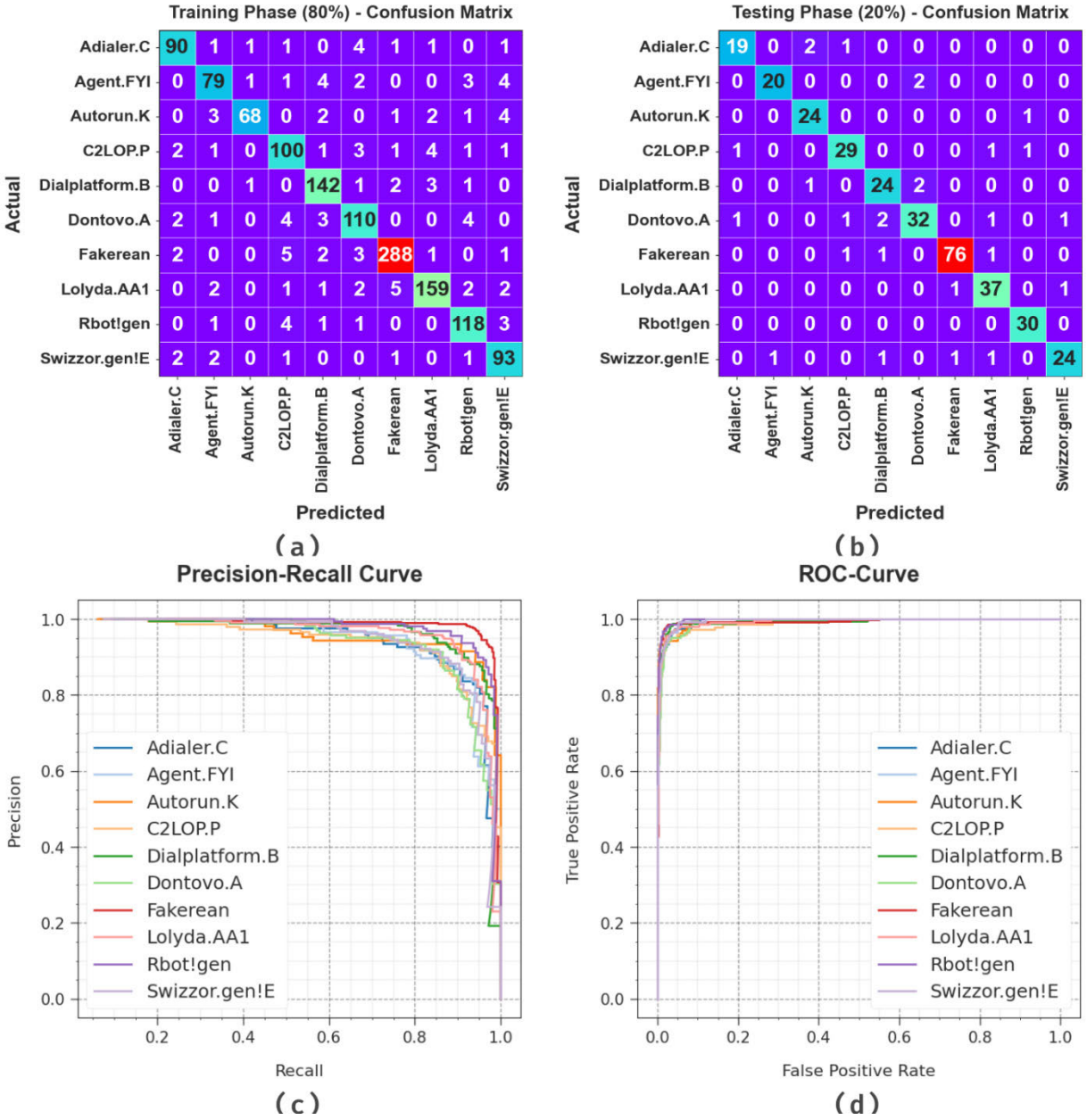

FIGURE 3. Classifier performance of (a-b) Confusion matrices and (c-d) PR and ROC curves.

removed from the wild. Every image in the dataset matches to an exact malware model and is resized to an even length for consistency in study. The dataset offers a valued resource for practitioners and researchers in order to progress and assess image-based malware recognition models and approaches. With its various range of malware samples and families, the Malimg dataset eases the survey of numerous features of image-based malware analysis, with feature extraction, detection and classification.

Fig. 3 displays the classifier performance of the SODCNN-IMC system under the test dataset. Figs. 3a-3b represents the confusion matrices accomplished by the SODCNN-IMC method at 80:20 of TRPH/TSPH. This figure denoted the SODCNN-IMC technique, which can be identified and categorized into ten classes. Next, Fig. 3c reveals the PR of the SODCNN-IMC system. This figure shows that the SODCNN-IMC technique achieves maximum PR performance. Lastly, Fig. 3d shows the ROC investigation of the SODCNN-IMC methodology. This result indicates that the SODCNN-IMC algorithm offers effective outcomes with maximum ROC values in diverse classes.

The malware classification performance of the SODCNN-IMC technique in the applied dataset is described in Table 2 and Fig. 4. These simulation values depict that the

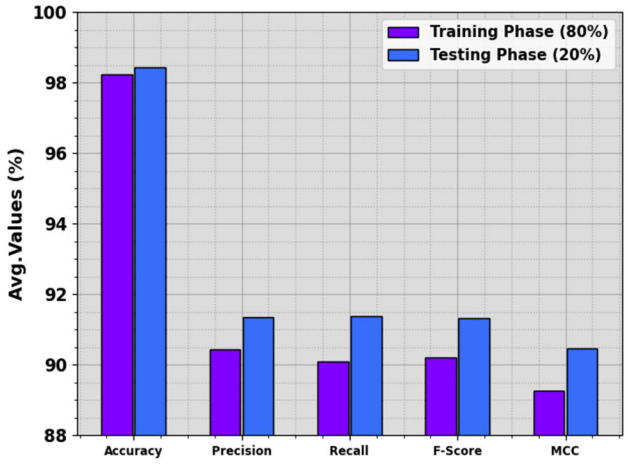

FIGURE 4. Average of SODCNN-IMC technique with 80:20 of TRPH/TSPH.

FIGURE 5. $Accu_y$ curve of the SODCNN-IMC methodology.

SODCNN-IMC technique achieves better performance with 10 class labels. With 80% of TRPH, the SODCNN-IMC technique gives an average $accu_y$ of 98.24%, $prec_n$ of 90.44%, $reca_l$ of 90.10%, F_{score} of 90.20%, and MCC of 89.26%. Additionally, with 20% of TSPH, the SODCNN-IMC technique offers an average $accu_y$ of 98.42%, $prec_n$ of 91.36%, $reca_l$ of 91.38%, F_{score} of 91.31%, and MCC of 90.46%.

The $accu_y$ curves for training (TR) and validation (VL) shown in Fig. 5 for the SODCNN-IMC technique offer valuable insights into its performance under various epochs. Mainly, it can be a consistent upgrading in both TR and TS $accu_y$ with increasing epochs, indicating the proficiency of the model in learning and recognizing designs from both TR and TS data. The upward trend in TS $accu_y$ underscores the model's adaptability to the TR dataset and its ability to make correct predictions under unnoticed data, emphasizing capabilities of persistent generalization.

Fig. 6 provides an extensive summary of the TR and TS loss values for the SODCNN-IMC technique over frequent epochs. This TR loss reliably reduces as a model improves weights to diminish classification errors below the datasets. The loss curves clearly illustrate the model's position with the

TABLE 2. Malware classifier analysis of SODCNN-IMC technique with 80:20 of TRPH/TSPH.

Classes	$Accu_y$	$Prec_n$	$Reca_l$	F_{Score}	MCC
TRPH (80%)					
Adialer. C	98.68	91.84	90.00	90.91	90.20
Agent.FYI	98.10	87.78	84.04	85.87	84.87
Autorun.K	98.83	95.77	83.95	89.47	89.07
C2LOP.P	97.73	85.47	87.72	86.58	85.35
Dialplatform.B	98.39	91.03	94.67	92.81	91.93
Dontovo.A	97.81	87.30	88.71	88.00	86.80
Fakerean	98.17	96.32	95.36	95.84	94.67
Lolyda.AA1	98.10	93.53	91.38	92.44	91.36
Rbot!gen	98.32	90.08	92.19	91.12	90.20
Swizzor.gen!E	98.32	85.32	93.00	89.00	88.18
Average	**98.24**	**90.44**	**90.10**	**90.20**	**89.26**
TSPH (20%)					
Adialer.C	98.54	90.48	86.36	88.37	87.62
Agent.FYI	99.12	95.24	90.91	93.02	92.58
Autorun.K	98.83	88.89	96.00	92.31	91.76
C2LOP.P	98.25	90.62	90.62	90.62	89.66
Dialplatform.B	97.95	85.71	88.89	87.27	86.18
Dontovo.A	97.08	88.89	84.21	86.49	84.89
Fakerean	98.54	97.44	96.20	96.82	95.87
Lolyda.AA1	98.25	90.24	94.87	92.50	91.54
Rbot!gen	99.42	93.75	100.00	96.77	96.51
Swizzor.gen!E	98.25	92.31	85.71	88.89	88.01
Average	**98.42**	**91.36**	**91.38**	**91.31**	**90.46**

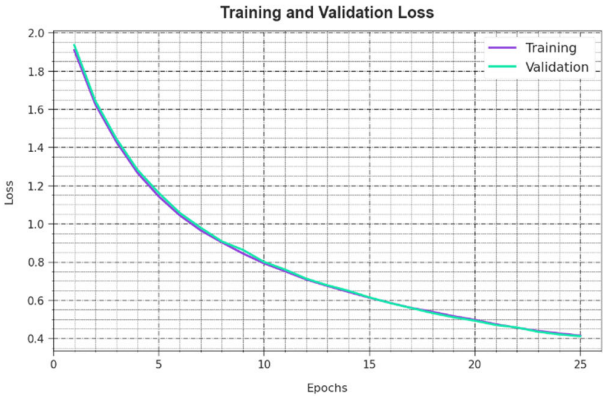

FIGURE 6. Loss curve of the SODCNN-IMC methodology.

TR data, emphasizing its ability to capture patterns effectively in both datasets. Noteworthy is the continuous refinement of parameters in the SODCNN-IMC methodology, aimed at lessening discrepancies between predictions and actual TR labels.

An extensive comparative study of the SODCNN-IMC method is provided with recent approaches [40] in Table 3.

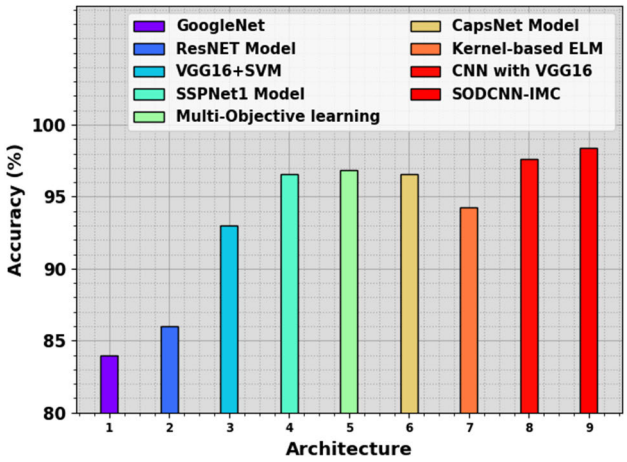

FIGURE 7. *Accu*$_y$ analysis of the SODCNN-IMC system with other models.

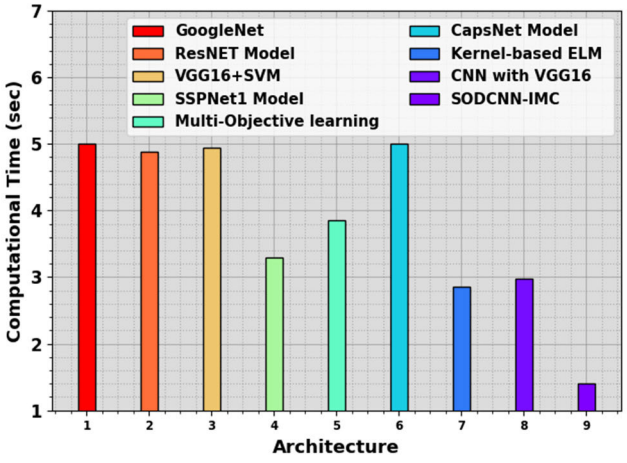

FIGURE 8. CT outcome of SODCNN-IMC technique with other systems.

TABLE 3. Comparison analysis of the SODCNN-IMC model with other models.

Architecture	Accuracy (%)	Computational Time (sec)
GoogleNet	84.00	5.01
ResNET Model	86.01	4.89
VGG16+SVM	92.97	4.94
SSPNet1 Model	96.60	3.30
Multi-Objective learning	96.86	3.85
CapsNet Model	96.58	5.01
Kernel-based ELM	94.25	2.85
CNN with VGG16	97.62	2.97
SODCNN-IMC	98.42	1.41

Fig. 7 represents a comparison result of the SODCNN-IMC approach in respect of $accu_y$. These accomplished findings stated that the SODCNN-IMC method has boosted effectiveness with increased $accu_y$ values. According to $accu_y$, the SODCNN-IMC technique offers enhanced $accu_y$ of 98.42% while the GoogleNet, ResNET, VGG16 + SVM, SSPNet1, Multi-Objective learning, CapsNet, Kernel-based ELM, and CNN with VGG16 methods obtain decreased $accu_y$ values of 84%, 86.01%, 92.97%, 96.60%, 96.86%, 96.58%, 94.25%, and 97.62%, respectively.

Fig. 8 illustrates a comparative review of the SODCNN-IMC technique with respect to CT. The obtained outcomes stated that the SODCNN-IMC algorithm gains increased performance with lesser CT values. According to CT, the SODCNN-IMC technique provides reduced CT of 1.41s while the GoogleNet, ResNET, VGG16 + SVM, SSPNet1, Multi-Objective learning, CapsNet, Kernel-based ELM, and CNN with VGG16 techniques obtain higher CT values of 5.01s, 4.89s, 4.94s, 3.30s, 3.85s, 5.01s, 2.85s, and 2.97s respectively.

The SODCNN-IMC method for malware image classification exhibits excellent performance because of its novel sequence of recent methods aimed particularly at this task. The combination of a ShuffleNet-based feature extraction offers a secure framework for proficiently capturing appropriate image features and optimizing computational resources without offering classifier accuracy. By leveraging the effectual channel shuffling functions and hierarchical feature extractor abilities of ShuffleNet, this method efficiently recognizes intricate patterns in the malware images, improving its discriminative power. In addition, the SO-based hyperparameter tuning process allows a fine-grained optimizer of model parameters, dynamically adjusting to the difficulties and nuances present in the database. This adaptive tuning process improves the method's flexibility and generalized ability, as it may result in enhanced solutions under various malware samples. Also, the combination of an ABiLSTM classification method enables the model to efficiently capture temporal dependencies and contextual data in order of feature extraction, added to refining the classification decisions. Overall, the SODCNN-IMC method excels in malware image classification by synergistically integrating effectual feature extractors, adaptive hyperparameter tuning, and refined sequence modelling processes because it leads to higher performance related to other methods.

V. CONCLUSION

In this research, we have recognized a new SODCNN-IMC methodology. The core concentration of the SODCNN-IMC methodology is to apply the hyperparameter-tuned DL method for the classification and detection of malware images. The SODCNN-IMC model contains three significant processes: ShuffleNet-based feature extractor, SO-based parameter tuning, and ABiLSTM-based classification. The design of the SO-based hyperparameter tuning process helps improve the overall recognition rate of the proposed model. The experimental evaluation of the SODCNN-IMC technique on the Malimg malware dataset demonstrates superior performance with a maximum accuracy of 98.42% over other

models. Therefore, the proposed model provides a promising avenue to enhance cybersecurity measures.

While SODCNN-IMC method provides major developments in image-based malware recognition, it is significant to recognize its limits. Initially, in spite of the efficacy gains attained over ShuffleNet and SO, the method might still face tasks in processing higher-resolution imageries or larger-scale datasets owing to characteristic computational limits. Moreover, the Abi-LSTM structure improves feature discrimination and representation, it may fight with taking very difficult spatial relationship or subtle designs within imageries, possibly foremost to misclassification or false negatives. Besides, the efficiency of SO model trusts deeply on the excellence and representativeness of the early parameter configuration, which may not constantly assurance optimum performance through every datasets or states.

Future work can focus on combining advanced methods like transfer learning (TL) and ensemble learning to improve the robustness and generalisation of these classification methods. Additionally, as cyber threats continue to develop, there may be a developing concentration on real-time recognition and response mechanisms leveraging image-based malware classification, allowing proactive defense against developing threats.

REFERENCES

[1] H. Nguyen, F. Di Troia, G. Ishigaki, and M. Stamp, "Generative adversarial networks and image-based malware classification," *J. Comput. Virol. Hacking Techn.*, vol. 19, no. 4, pp. 579–595, Feb. 2023, doi: 10.1007/s11416-023-00465-2.

[2] R. Mitsuhashi and T. Shinagawa, "Deriving optimal deep learning models for image-based malware classification," in *Proc. 37th ACM/SIGAPP Symp. Appl. Comput.*, Apr. 2022, pp. 1727–1731, doi: 10.1145/3477314.3507242.

[3] P. Prajapati and M. Stamp, "An empirical analysis of image-based learning techniques for malware classification," in *Malware Analysis Using Artificial Intelligence and Deep Learning*. Cham, Switzerland: Springer, 2020, pp. 411–435, doi: 10.1007/978-3-030-62582-5.

[4] C. Reilly, S. O Shaughnessy, and C. Thorpe, "Robustness of image-based malware classification models trained with generative adversarial networks," in *Proc. Eur. Interdiscipl. Cybersecurity Conf.*, Jun. 2023, pp. 92–99, doi: 10.1145/3590777.3590792.

[5] A. Fathurrahman, A. Bejo, and I. Ardiyanto, "Lightweight convolution neural network for image-based malware classification on embedded systems," in *Proc. Int. Seminar Mach. Learn., Optim., Data Sci. (ISMODE)*, 2022, pp. 12–16, doi: 10.1109/ismode53584.2022.9743111.

[6] Y. X. M. Tan, K. T. Yew, and L. J. Sern, "MalwareCLIP: Towards a scalable and explainable image-based malware classification," in *Proc. Int. Conf. Commun., Comput., Cybersecurity, Informat. (CCCI)*, Oct. 2023, pp. 1–8, doi: 10.1109/ccci58712.2023.10290765.

[7] D. Pant and R. Bista, "Image-based malware classification using deep convolutional neural network and transfer learning," in *Proc. 3rd Int. Conf. Adv. Inf. Sci. Syst.*, Nov. 2021, pp. 1–9, doi: 10.1145/3503047.3503081.

[8] T. H. Hai, V. Van Thieu, T. T. Duong, H. H. Nguyen, and E.-N. Huh, "A proposed new endpoint detection and response with image-based malware detection system," *IEEE Access*, vol. 11, pp. 122859–122875, 2023, doi: 10.1109/ACCESS.2023.3329112.

[9] V. S. Jeyalakshmi, N. Krishnan, and J. Jayapriya, "Deep convolutional neural networks network with transfer learning for image-based malware analysis," in *Proc. Int. Conf. Innov. Comput. Commun.*, 2023, pp. 39–51, doi: 10.1007/978-981-99-3010-4_4.

[10] A. Santone, "Assessing deep learning predictions in image-based malware detection with activation maps," in *Proc. 18th Int. Workshop, Secur. Trust Manag. (STM)*, Copenhagen, Denmark, vol. 13867, 2023, p. 104, doi: 10.1007/978-3-031-29504-1_6.

[11] F. Wang, X. Shi, F. Yang, R. Song, Q. Li, Z. Tan, and C. Wang, "MalSort: Lightweight and efficient image-based malware classification using masked self-supervised framework with Swin transformer," *J. Inf. Secur. Appl.*, vol. 83, Jun. 2024, Art. no. 103784.

[12] K. E. Ketebu, G. O. Onwodi, K. E. Ukhurebor, B. M. Eneche, and N. K. Yaah-Nyakko, "A recent survey of image-based malware classification using convolution neural network," *J. Auto. Intell.*, vol. 7, no. 5, p. 1287, Mar. 2024.

[13] K. Jaisinghani and S. Singh, "Recent advances in image based malware classification through the lens of deep learning—A systematic literature review," *Int. J. Intell. Syst. Appl. Eng.*, vol. 11, no. 9s, pp. 414–423, 2023.

[14] H. Al-Qadasi, D. Y. M. Benchadi, S. Chehida, K. Fukui, and S. Bensalem, "Neural network innovations in image-based malware classification: A comparative study," in *Proc. Int. Conf. Adv. Inf. Netw. Appl.* Cham, Switzerland: Springer Nature, 2024, pp. 252–265.

[15] S. Yaseen, M. M. Aslam, M. Farhan, M. R. Naeem, and A. Raza, "A deep learning-based approach for malware classification using machine code to image conversion," *Tech. J.*, vol. 28, no. 1, pp. 36–46, 2023.

[16] T. T. Son, C. Lee, H. Le-Minh, N. Aslam, and V. C. Dat, "An enhancement for image-based malware classification using machine learning with low dimension normalized input images," *J. Inf. Secur. Appl.*, vol. 69, Sep. 2022, Art. no. 103308.

[17] S. O'Shaughnessy and S. Sheridan, "Image-based malware classification hybrid framework based on space-filling curves," *Comput. Secur.*, vol. 116, May 2022, Art. no. 102660.

[18] P. Yadav, N. Menon, V. Ravi, S. Vishvanathan, and T. D. Pham, "A two-stage deep learning framework for image-based Android malware detection and variant classification," *Comput. Intell.*, vol. 38, no. 5, pp. 1748–1771, Oct. 2022.

[19] Z. Zhao, D. Zhao, S. Yang, and L. Xu, "Image-based malware classification method with the AlexNet convolutional neural network model," *Secur. Commun. Netw.*, vol. 2023, pp. 1–15, Apr. 2023, doi: 10.1155/2023/6390023.

[20] Y. Jian, H. Kuang, C. Ren, Z. Ma, and H. Wang, "A novel framework for image-based malware detection with a deep neural network," *Comput. Secur.*, vol. 109, Oct. 2021, Art. no. 102400, doi: 10.1016/j.cose.2021.102400.

[21] R. Chaganti, V. Ravi, and T. D. Pham, "Image-based malware representation approach with EfficientNet convolutional neural networks for effective malware classification," *J. Inf. Secur. Appl.*, vol. 69, Sep. 2022, Art. no. 103306, doi: 10.1016/j.jisa.2022.103306.

[22] H. Polsani and H. Jiang, "DeepGray: A novel approach to malware classification using grayscale images with deep learning," 2023, doi: 10.17605/OSF.IO/EADBM.

[23] K. Bakour and H. M. Ünver, "DeepVisDroid: Android malware detection by hybridizing image-based features with deep learning techniques," *Neural Comput. Appl.*, vol. 33, no. 18, pp. 11499–11516, Sep. 2021, doi: 10.1007/s00521-021-05816-y.

[24] B. Zou, C. Cao, F. Tao, and L. Wang, "IMCLNet: A lightweight deep neural network for image-based malware classification," *J. Inf. Secur. Appl.*, vol. 70, Nov. 2022, Art. no. 103313, doi: 10.1016/j.jisa.2022.103313.

[25] A. Copiaco, L. El Neel, T. Nazzal, H. Mukhtar, and W. Obaid, "A neural network approach to a grayscale image-based multi-file type malware detection system," *Appl. Sci.*, vol. 13, no. 23, p. 12888, Nov. 2023, doi: 10.3390/app132312888.

[26] I. Obaidat, M. Sridhar, K. M. Pham, and P. H. Phung, "Jadeite: A novel image-behavior-based approach for Java malware detection using deep learning," *Comput. Secur.*, vol. 113, Feb. 2022, Art. no. 102547, doi: 10.1016/j.cose.2021.102547.

[27] V. Ravi and M. Alazab, "Attention-based convolutional neural network deep learning approach for robust malware classification," *Comput. Intell.*, vol. 39, no. 1, pp. 145–168, Feb. 2023.

[28] R. Chaganti, V. Ravi, and T. D. Pham, "A multi-view feature fusion approach for effective malware classification using deep learning," *J. Inf. Secur. Appl.*, vol. 72, Feb. 2023, Art. no. 103402.

[29] B. A. Ouahab, I. L. Elaachak, and M. Bouhorma, "Image-based malware classification using multi-layer perceptron," in *Networking, Intelligent Systems and Security*. Singapore: Springer, 2022, pp. 453-464.

[30] N. Ullah, J. Khan, S. El-Sappagh, N. El-Rashidy, and M. Khan, "A holistic approach to identify and classify COVID-19 from chest radiographs, ECG, and CT-scan images using ShuffleNet convolutional neural network," *Diagnostics*, vol. 13, no. 1, p. 162, Jan. 2023, doi: 10.3390/diagnostics13010162.

[31] Q. Abu Al-Haija, "Leveraging ShuffleNet transfer learning to enhance handwritten character recognition," *Gene Expression Patterns*, vol. 45, Sep. 2022, Art. no. 119263.
[32] S. Ghosh, M. J. Mondal, S. Sen, S. Chatterjee, N. K. Roy, and S. Patnaik, "A novel approach to detect and classify fruits using ShuffleNet V2," in *Proc. IEEE Appl. Signal Process. Conf. (ASPCON)*, Oct. 2020, pp. 163–167.
[33] R. A. Khurma, E. Alhenawi, M. Braik, F. A. Hashim, A. Chhabra, and P. A. Castillo, "A bio-medical snake optimizer system driven by logarithmic surviving global search for optimizing feature selection and its application for disorder recognition," *J. Comput. Design Eng.*, vol. 10, no. 6, pp. 2361–2383, Nov. 2023, doi: 10.1093/jcde/qwad101.
[34] F. A. Hashim and A. G. Hussien, "Snake optimizer: A novel meta-heuristic optimization algorithm," *Knowledge-Based Syst.*, vol. 242, Apr. 2022, Art. no. 108320.
[35] H. Fu, H. Shi, Y. Xu, and J. Shao, "Research on gas outburst prediction model based on multiple strategy fusion improved snake optimization algorithm with temporal convolutional network," *IEEE Access*, vol. 10, pp. 117973–117984, 2022.
[36] M. Seydali, F. Khunjush, B. Akbari, and J. Dogani, "Cbs: A deep learning approach for encrypted traffic classification with a mixed spatio-temporal and statistical features classification," *SSRN Electron. J.*, 2022, doi: 10.1109/ACCESS.2022.3220765.
[37] J. Guo, M. Liu, P. Luo, X. Chen, H. Yu, and X. Wei, "Attention-based BILSTM for the degradation trend prediction of lithium battery," *Energy Rep.*, vol. 9, pp. 655–664, Apr. 2023.
[38] J. Zhang, Y. Liu, and H. Yuan, "Attention-based residual BiLSTM networks for human activity recognition," *IEEE Access*, vol. 11, pp. 94173–94187, 2023.
[39] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images," in *Proc. 8th Int. Symp. Visualizat. for Cyber Secur.*, Jul. 2011, pp. 94173–94187, doi: 10.1145/2016904.2016908.
[40] M. J. Awan, O. A. Masood, M. A. Mohammed, A. Yasin, A. M. Zain, R. Damaševičius, and K. H. Abdulkareem, "Image-based malware classification using VGG19 network and spatial convolutional attention," *Electronics*, vol. 10, no. 19, p. 2444, Oct. 2021, doi: 10.3390/electronics10192444.

SALAHALDEEN DURAIBI received the B.S. degree in computer science from Jazan University, Saudi Arabia, the M.S. degree in computer science from Kentucky State University, USA, and the Ph.D. degree in computer science from the University of Idaho, USA. Currently, he is an Assistant Professor with the College of Engineering and Computer Science, Jazan University. His research interests include computer and network security, intrusion detection, AI, and ML.